UNIVERSITÉ DE LA MANOUBA, TUNISIE
ÉCOLE NATIONALE DES SCIENCES DE L'INFORMATIQUE

UNIVERSITÉ BORDEAUX I, FRANCE
SCIENCES & TECHNOLOGIES

EN COTUTELLE INTERNATIONALE

THÈSE

présentée en vue de l'obtention du titre de

DOCTEUR EN INFORMATIQUE

par

Faten BEN HMIDA

Titre

ÉVALUATION DES PERFORMANCES DES SYSTÈMES MULTI-AGENTS

Réalisée aux laboratoires

Laboratoire Stratégies d'Optimisation et Informatique intelligentE, Tunisie

Laboratoire de l'Intégration du Matériau au Système, France

Soutenue à l'ENSI (date prévisionnelle 17/12/2013)

JURY D'EXAMEN

Mme. Henda BEN GHEZELA	Professeur, Université La Manouba	Présidente du jury
M. Bernard ARCHIMÈDE	Professeur, Université de Toulouse	Rapporteur
M. Sadok BOUAMAMA	Maître de conférences HDR, Université La Manouba	Rapporteur
M. Moncef TAGINA	Professeur, Université La Manouba	Directeur de thèse
M. Rémy DUPAS	Professeur, Université Bordeaux I	Directeur de thèse

Évaluation des Performances des Systèmes Multi-Agents

Résumé : Cette thèse s'intéresse à la question de l'évaluation des Systèmes Multi-Agents (SMA). Les caractéristiques propres que possèdent ces derniers, notamment en termes d'autonomie, de distribution, de dynamique et de socialité, ont grandement contribué à l'élargissement de leurs champs d'application, mais en contrepartie, elles ont rendu leur analyse plus ardue. Ainsi, les méthodes d'évaluation dans les systèmes informatiques classiques s'avèrent insuffisantes à analyser les SMA, étant donné qu'elles ne tiennent pas compte de leurs spécificités. L'objectif de cette thèse consiste donc à proposer une approche générique pour l'évaluation des SMA en se basant sur la mesure de leurs caractéristiques fonctionnelles. A cet effet, le besoin de disposer d'informations sur l'exécution du système à évaluer est manifeste. C'est dans ce cadre qu'une nouvelle approche d'observation des SMA est proposée. Les résultats de ces observations sont exploités pour construire une abstraction du système sous forme d'un modèle, lequel est étudié pour définir les mesures de performances. L'analyse se focalise sur deux caractéristiques essentielles, à la base de la dynamique et de la socialité des SMA : la communication et l'organisation. Les expérimentations de la solution proposée portent sur deux applications multi-agents. La première est une application de diagnostic des pannes dans un environnement industriel et la seconde est une application de pilotage et de gestion de la production dans les chaînes logistiques.

Mots clés : Systèmes multi-agents, évaluation des performances, observation, modélisation, mesure, communication, organisation, théorie des graphes.

Performance Evaluation of MultiAgent Systems

Abstract: This thesis focuses on the issue of MultiAgent Systems (MAS) evaluation. The MAS own characteristics, namely autonomy, distribution, dynamicity and sociality, have greatly contributed to the expansion of their application scope; but in return they made their analysis more difficult. Thus, evaluation methods in classic computer systems are insufficient to analyse MAS, since they do not take into account their specificities. The objective of this thesis is to provide a generic approach for the evaluation of MAS by measuring their functional characteristics. To this end, the need for information about the execution of the system to be evaluated is evident. In this context, a new approach to observe MAS is proposed. The results of these observations are exploited to build an abstraction model of the system which is studied in order to define performance metrics. The analysis focuses on two key characteristics, at the basis on the dynamics and sociality in MAS: communication and organization. The experiments of the proposed solution are performed on two multiagent applications. The first is an application of fault diagnosis in an industrial environment and the second is an application of control and production planning in supply chains.

Keywords: Multiagent systems, performance evaluation, observation, modelling, measure, communication, organization, graph theory.

Remerciements

C'est avec immense plaisir et grande émotion que j'aborde l'écriture de ces lignes qui marquent la fin de cette première étape de mon parcours de recherche et certainement le début d'une autre que j'espère toute aussi fructueuse en résultats et riche en rencontres. Je tiens à travers ces remerciements à témoigner de ma plus grande reconnaissance envers toutes les personnes qui, de près ou de loin, ont contribué à l'achèvement de ce travail.

Je remercie en premier lieu Madame Henda Hajjami Ben Ghezela, Professeur à l'Université de La Manouba, de l'honneur qu'elle m'a accordé en acceptant d'examiner ma thèse et de présider le jury de ma soutenance.

Je remercie également Monsieur Bernard Archimède, Professeur à l'Ecole Nationale des Ingénieurs de Tarbes et Monsieur Sadok Bouamama, Maître de Conférences HDR à l'Université de La Manouba, d'avoir accepté d'évaluer mon travail et d'être les rapporteurs de cette thèse.

J'adresse mes plus vifs remerciements à mon directeur de thèse Monsieur Moncef Tagina, Professeur à l'Université de La Manouba, qui m'a accompagnée de près tout au long de mon parcours de recherche. Je le remercie pour sa disponibilité, ses conseils avisés, son encouragement inlassable et son soutien solide dans les moments critiques. Ça a été et ce sera toujours un plaisir et un honneur que de travailler avec lui.

Je remercie également mon co-directeur de thèse Monsieur Rémy Dupas, Professeur à l'Université Bordeaux I, de m'avoir accueillie au sein de son équipe PSP au laboratoire IMS de Bordeaux, et de m'avoir inconditionnellement soutenue dans toutes mes démarches. Je le remercie aussi de m'avoir accordé une grande liberté dans le travail tout en assurant un suivi méticuleux.

Mes plus chaleureux remerciements s'adressent à ma co-encadrante de thèse Mme Wided Lejouad Chaari grâce à qui mon intérêt à la recherche scientifique s'est éveillé. Elle m'a accompagnée dès le début de mon parcours. C'est notamment grâce à son cours sur les systèmes multi-agents que s'est développée ma motivation à travailler dans ce domaine. Je lui dois beaucoup et je tiens à lui exprimer ma reconnaissance de m'avoir généreusement transmis son savoir et savoir-faire. Je la remercie sincèrement pour son suivi implacable de mes travaux, pour sa confiance, et pour tout le temps et l'énergie qu'elle a su me consacrer durant ces dernières années. C'est une chance que de pouvoir travailler avec une personne ayant ses qualités scientifiques et humaines.

Je remercie également Anne Séguy Garcia, maître de conférences à l'Université Bordeaux I, pour sa participation active et son implication dans le suivi de cette thèse. Je lui suis reconnaissante pour son aide en productique et pour ses relectures du manuscrit.

Une pensée particulière s'adresse à Julien François, maître de conférences à l'Université Bordeaux I, qui n'a pas hésité à me prodiguer son aide chaque fois que j'ai sollicité ses compétences.

Je ne laisserai pas cette occasion passer sans exprimer ma sympathie envers toute l'équipe du département QLIO de l'Université Bordeaux I. Je les remercie de m'avoir chaleureusement accueillie et d'avoir contribué à rendre mon séjour parmi eux si riche et agréable. Un grand merci en particulier à Marie-Hélène et à Fréderic pour leurs amicales attentions. Merci aussi à Pascale, Séverine, Vincent, Danielle et tous les autres que j'aurais involontairement oublié de citer. Leur rencontre m'a été très enrichissante.

Je suis également très redevable envers mes chers amis Olfa et Hafedh Saidani pour leur hospitalité et leur soutien moral et matériel incontestable durant mes périodes de mobilité. Merci aussi à Gaelle et Ali Dardour pour leur accueil et la bonne ambiance de famille qui nous manque tant quand on est loin des siens. Merci à Fany Viellard et sa famille de m'avoir offert leur précieuse amitié, et à Mouna pour le partage de moments inoubliables.

Je ne manquerai pas à cette occasion d'adresser mes remerciements chaleureux à mes collègues de l'ENSI et du laboratoire SOIE, particulièrement à ma chère amie de toujours Raoudha pour sa présence inestimable, à mes amis Ons, Imtiez, Ramla et Haythem pour leur aide précieuse et grande disponibilité. A Mariem, Olfa, Nessrine, Houda, Aroua, Nour et Salma pour leur amitié, affection et partage.

J'exprime également ma reconnaissance envers toutes les personnes qui ont contribué à ma formation, à mes enseignants de l'ENSI. Je suis particulièrement redevable à Madame Leila Azouz Saidane pour son soutien inestimable.

Je profite enfin de cet espace pour adresser l'expression de ma profonde gratitude à mes proches… A Maman, jamais je ne pourrais m'acquitter de ma dette envers toi. Je te dois tout. Puisse l'accomplissement de ce travail t'apporter satisfaction... Papa, merci de m'avoir toujours soutenue et encouragée à aller le plus loin possible dans mes études… A mes sœurs Mariem et Sarra et ma tante Saida, merci pour votre affection et de m'avoir supportée quand ça allait moins bien… A mes petits neveux Youssef et Yessine, merci d'avoir égayé ma vie et l'avoir rendu plus belle… A mon oncle Hédi, grâce à toi j'ai appris l'amour du savoir et le sens du dévouement dans le travail… A feu mon oncle Mohamed, tu nous as quitté si tôt sans voir l'achèvement de ce travail que tu attendais tant… Que Dieu te comble de son infinie miséricorde… A feu mes grands parents, vous serez à jamais dans mon cœur et l'amour dont vous m'avez comblé guidera toujours mes pas dans cette vie… A toute ma famille et mes amis, je dis merci d'avoir été toujours là et d'avoir cru en moi, je vous dédie ce travail en guise d'amour et de reconnaissance.

TABLE DES MATIÈRES

TABLE DES ILLUSTRATIONS

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GENERALE

L'intelligence Artificielle (IA) est une branche des sciences de l'informatique qui s'intéresse à la reproduction de certains aspects de l'intelligence humaine dans le but de développer des algorithmes, des méthodologies et des systèmes destinés à la résolution de problèmes complexes. L'approche classique de l'IA aborde cette question d'une manière centralisée, ainsi un seul processus monopolise la gestion des connaissances, la manipulation des ressources et la réalisation des tâches. Cette approche centralisée a été remise en question vu les inconvénients qu'elle présente, en particulier la nécessité d'intégrer au sein d'une même entité des connaissances et des compétences diverses et nombreuses, qui dans la réalité appartiendraient à plusieurs individus collaborant ensemble pour réaliser un but commun. De ce constat est né le besoin de passer d'une vision purement individuelle de la résolution d'un problème à une vision collective qui nécessite la distribution de l'intelligence sur plusieurs entités. C'est ainsi qu'est apparue l'Intelligence Artificielle Distribuée (IAD). L'IAD a elle-même introduit un nouveau paradigme, il s'agit du paradigme multi-agents. Dans ce dernier, on s'intéresse à la manière de répartir un problème sur un certain nombre d'entités autonomes appelées agents, capables d'agir dans un environnement déterminé, de communiquer, de coopérer et de résoudre les éventuels conflits dans le but de réaliser un objectif commun.

Les Systèmes Multi-Agents (SMA) ont, depuis, occupé une place importante au cœur des nouvelles technologies, et se sont positionnés au carrefour de plusieurs autres disciplines connexes où ils puisent leurs sources d'inspiration comme la sociologie, la philosophie, l'éthologie, les réseaux, etc. Ils s'avèrent d'une grande efficacité dans des champs d'application variés où les approches de développement classiques restent limitées. Les SMA sont particulièrement adaptés aux systèmes complexes, ouverts, distribués, asynchrones et dynamiques. Ils sont utilisés dans plusieurs applications comme la production industrielle, la supervision de processus, la robotique, le diagnostic, le trafic aérien, le commerce électronique, la recherche d'informations, etc.

Problématique

Malgré l'élargissement de leurs champs d'utilisation et la croissance continuelle des applications qu'ils permettent de réaliser, les SMA souffrent toujours de limitations considérables en matière d'évaluation des performances. Les caractéristiques propres que possèdent ces systèmes ont, certes, contribué largement à leurs succès mais en contre partie ont rendu leur analyse plus ardue. En effet, les approches d'évaluation classiques, qui considèrent le système analysé comme une boite noire et qui se limitent à des critères de performance usuels s'avèrent insuffisants pour les SMA. Ainsi, jusqu'à l'heure actuelle, on ne dispose pas d'outil ou de méthodologie pour évaluer convenablement un SMA ou comparer deux SMA entre eux. Cette question n'a été que peu posée et il n'existe pas beaucoup de travaux qui s'y sont intéressés. L'étude et la définition des critères d'évaluation et de comparaison des SMA demeurent des problèmes ouverts et font encore l'objet de recherches au sein de la communauté scientifique afférente. De plus, la plupart des travaux existants dans la littérature s'intéressent principalement à l'évaluation des méthodologies de conception et des plates-formes de développement et ne couvrent pas les applications multi-agents développées. Les résultats étant, dans ce cadre, dépendant d'applications particulières et parfois très spécifiques à des scénarios préétablis, et ne peuvent donc pas être généralisés. C'est dans ce cadre que s'inscrit notre thèse dont l'objectif est de définir une approche d'évaluation des SMA qui prenne en considération leurs particularités et qui couvre leur aspect applicatif.

Plan du manuscrit de thèse

Le présent manuscrit de thèse s'articule autour de quatre chapitres.

Le premier chapitre introduit dans un premier temps les concepts de base du domaine multi-agents et expose les principales caractéristiques des SMA. Il présente ensuite l'état de l'art relatif à la question de l'évaluation dans le paradigme agent. Dans ce contexte les principaux travaux de référence se déclinent en plusieurs catégories, à savoir : l'évaluation de la technologie agent, l'évaluation des approches de modélisation et des méthodologies de conception, l'évaluation des outils de développement et des plates-formes d'exécution, la validation et la vérification de modèles et enfin l'évaluation des applications et des systèmes multi-agents développés. A la fin de ce chapitre, la problématique est soulignée par la synthèse et la critique de l'état de l'art.

Le deuxième chapitre présente l'approche proposée pour la résolution du problème de l'évaluation des SMA. Il décrit d'abord, l'architecture globale du système d'évaluation

proposé et se concentre ensuite sur la phase d'observation. En effet, l'observation est une étape indispensable pour appréhender le comportement du SMA évalué, elle permet de collecter les informations pertinentes sur son exécution et de générer les traces en vue de leur analyse ultérieure. Dans ce cadre, les notions essentielles relatives à l'observation dans les systèmes informatiques en général et dans les SMA en particulier, sont introduites. Ensuite, la conception et le fonctionnement de la solution d'observation proposée sont détaillés.

Le troisième chapitre décrit l'approche de mesure des performances des SMA proposée. A cet effet, une classification des caractéristiques des SMA est dégagée. Cette classification permet de guider le choix des critères d'évaluation ainsi que le modèle qui permettra d'analyser ces derniers. Le choix des graphes est ensuite expliqué et justifié et, en se basant sur la théorie des graphes, un ensemble de mesures est proposé pour l'évaluation des caractéristiques choisies.

Le quatrième chapitre présente les résultats de mise en œuvre et de test de la solution d'évaluation proposée. Pour ce faire, deux applications multi-agents sont proposées. La première est une application de diagnostic des pannes dans un système industriel. La seconde est une application de gestion de production et de pilotages des chaines logistiques. Pour chacune de ces applications, les valeurs des critères mesurés sont présentées et interprétées.

Le présent manuscrit est clôturé par une conclusion et quelques perspectives intéressantes à explorer en continuité de cette thèse.

CHAPITRE 1

ÉVALUATION DES SYSTÈMES MULTI-AGENTS ÉTAT DE L'ART

Sommaire

Ce premier chapitre présente une synthèse de l'état de l'art sur les SMA et sur leur évaluation. Il se compose de deux parties. Dans la première nous introduisons le domaine des systèmes multi-agents à travers la revue de quelques notions de base. Nous commençons par présenter quelques définitions, ensuite nous passons en revue les caractéristiques des SMA pour enchaîner avec la présentation des supports d'aide à la conception et au développement de ces systèmes tels que les méthodologies de conception et les plates-formes de développement. Dans la deuxième partie de ce chapitre, nous exposons les travaux réalisés autour de la thématique de l'évaluation dans le cadre des SMA. Nous proposons à cet effet une classification de ces travaux selon l'aspect visé par l'évaluation : général, conceptuel, applicatif, etc. Une critique de l'existant et une synthèse de la problématique seront présentées à la fin de ce chapitre.

1. Les systèmes multi-agents

Les SMA sont devenus, depuis quelques années, un domaine de recherche en pleine effervescence et suscitant de plus en plus d'intérêt parmi la communauté scientifique. Il s'agit d'un paradigme assez récent des sciences et technologies de l'information et de la communication, principalement issu du domaine de l'IA mais qui puise aussi ses sources d'inspiration de plusieurs autres disciplines telles que l'informatique répartie, les sciences cognitives, la sociologie, etc. Par ailleurs, le domaine des SMA s'articule autour de nouvelles notions telles que l'autonomie, la décentralisation et l'interaction. Nous présentons dans ce qui suit une synthèse de l'état de l'art sur les SMA en commençant par les notions de base.

1.1. Définitions

Il existe un nombre important d'ouvrages qui s'intéressent aux agents et aux systèmes multi-agents, différentes définitions du terme agent y sont proposées, cependant, aucune n'est acceptée à l'unanimité par la communauté scientifique afférente, et il n'existe pas encore de définition consensuelle du terme agent.

Pour cette raison, nous ne nous limitons pas ici à une unique définition d'un agent mais nous présentons celles qui nous semblent mieux résumer cette notion. D'après Wooldridge & Jennings [Woo02] :

« *Un agent est une entité informatique située dans un environnement et capable d'agir dessus d'une manière autonome dans le but d'atteindre les objectifs pour lesquels elle a été conçue.* »

Une autre définition du terme agent est donnée par Ferber [Fer95] :

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

A partir de ces définitions, nous pouvons dégager une propriété spécifique des agents, à savoir l'autonomie qui signifie l'indépendance et la liberté d'action en dehors de contrôle extérieur. Cette particularité des agents engendre d'autres caractéristiques bien propres aux SMA et qui font toute leur différence avec les systèmes informatiques classiques. Ces caractéristiques seront dégagées et présentées dans le paragraphe suivant.

Une autre propriété des agents réside dans la flexibilité de leurs comportements vis-à-vis de leur environnement [Woo02]. La flexibilité se traduit par :

- La réactivité : c'est la capacité de l'agent à percevoir son environnement et à réagir aux changements qui y surgissent, dans le but de satisfaire ses objectifs.
- La proactivité : c'est la prise d'initiative d'agir en vue de réaliser un but et non seulement la réponse immédiate à des évènements particuliers.
- La socialité : c'est la capacité de l'agent à interagir avec d'autres agents. La coopération, la négociation, la communication et le partage de buts sont des aspects de la socialité de l'agent.

Mises à part ces propriétés communes aux agents, ces derniers peuvent être très variés. On distingue, par exemple, les agents stationnaires et les agents mobiles : un agent stationnaire s'exécute sur une même machine alors qu'un agent mobile peut se déplacer en transportant son code et ses données à travers un réseau informatique pour les exécuter dans un site distant.

On distingue aussi les agents réactifs et les agents cognitifs ou intelligents : leur différence se situe au niveau de leurs architectures internes et de la manière dont les informations perçues par ces agents sont traitées. En effet, le comportement des agents réactifs est régi par des réponses immédiates à des stimuli fournis par l'environnement, alors que le comportement des agents cognitifs se base plutôt sur leur capacité à représenter symboliquement leurs environnements, à mémoriser leurs états passés et à planifier leurs actions en tenant compte de leurs connaissances, de leurs croyances et de leurs intentions.

Maintenant que nous avons présenté la notion d'agent, la définition d'un SMA s'impose. D'après Ferber [Fer95] :

« *On peut définir un système multi-agents comme un système composé des éléments suivants:*

- *un environnement (par exemple un espace disposant d'une métrique) ;*
- *un ensemble d'objets situés, c'est-à-dire auxquels on peut associer une position dans l'environnement ;*
- *un ensemble d'agents, qui sont des objets particuliers (les agents diffèrent des autres objets par leur capacité à agir sur eux mêmes et sur l'environnement) ;*
- *un ensemble de relations, ou de contraintes, qui unit des objets et/ou des agents entre eux ;*
- *un ensemble d'opérations déterminant les façons dont les agents peuvent agir sur les objets ;*
- *un ensemble d'opérateurs représentant la façon dont ces actions sont effectuées et leur effet sur l'environnement.* »

La philosophie des SMA repose sur le principe de faire interagir plusieurs agents entre eux pour réaliser un objectif global. Chacun de ces agents détient une représentation partielle de son environnement, et possède des connaissances et des compétences propres qui lui permettent d'évoluer dans l'environnement commun de manière à satisfaire son but local. La figure 1.1 illustre la structure typique d'un SMA.

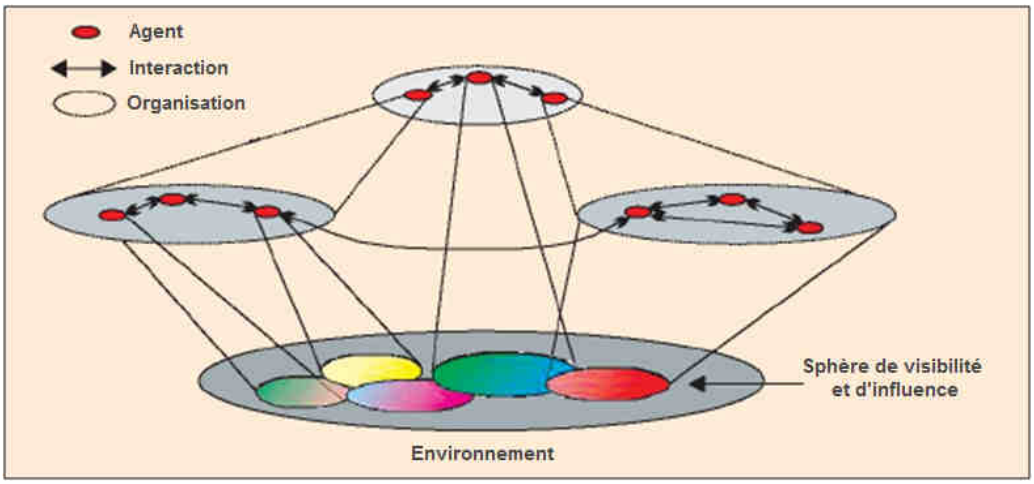

Figure 1.1 : Structure typique d'un système multi-agents [Woo02]

Les agents du SMA sont regroupés en relations organisationnelles et ont chacun "une sphère d'influence", c'est-à-dire que chaque agent possède le contrôle d'une partie de l'environnement. Ces sphères d'influence peuvent parfois interférer et c'est ce qui engendre les situations d'interactions entre les agents des diverses organisations.

La force du paradigme multi-agents provient de la flexibilité et de la variété des types de ces interactions et des modèles d'organisation impliqués dans de tels systèmes. Par ailleurs, les SMA doivent leurs succès à bien d'autres raisons, principalement du fait qu'ils répondent à certains besoins et qu'ils présentent certaines caractéristiques fonctionnelles qu'on ne retrouve pas forcément dans les autres technologies telles que la technologie orientée-objet.

1.2. Caractéristiques des systèmes multi-agents

L'objectif de ce paragraphe est de présenter une vue d'ensemble sur les différentes caractéristiques des SMA. Il est important à ce stade de distinguer entre les caractéristiques fonctionnelles des SMA et les besoins non fonctionnels auxquels ils peuvent répondre tels que l'extensibilité, la portabilité, la fiabilité ou la robustesse [Boi04]. Selon les applications, ces caractéristiques ne sont pas forcément toutes nécessaires. Nous en présentons dans ce qui suit une liste en nous référant à une étude effectuée dans [Boi04] :

- **L'autonomie**

L'autonomie désigne communément l'indépendance et la capacité d'action et de prise de décision. Dans les SMA, l'autonomie peut se définir en trois points essentiels : l'existence propre et indépendante de l'agent, le maintien de sa viabilité en dehors de contrôle extérieur et la prise de décision en tenant compte uniquement de ses perceptions et de ses connaissances. En effet, les agents sont capables de planifier leurs actions, de raisonner et de résoudre des problèmes sans contrôle extérieur.

- **La distribution**

La distribution dans les SMA est une caractéristique qui répond au besoin de distribution physique des connaissances et des traitements. Cette distribution impose un découpage modulaire dont la nécessité est parfois accentuée par l'absence de modèle global du problème à résoudre. Dans un environnement multi-agents, la distribution signifie donc que plusieurs agents participent à la réalisation d'un objectif global en se partageant les connaissances, les traitements, les tâches et les ressources.

- **La décentralisation**

La décentralisation signifie la répartition du contrôle. L'éventuelle complexité du problème à résoudre rend difficile à l'utilisateur de gérer le contrôle total du système. Ainsi, une approche décentralisée de résolution consiste à léguer à chaque agent une

partie de ce dernier. La décentralisation peut être dictée, entre autres, par des contraintes liées à la distribution physique du système ou par des limitations des capacités de décision globale.

- **La communication**

La communication permet aux agents d'échanger des informations et assure ainsi la cohérence du comportement global du système malgré la décentralisation. Il existe deux moyens de communication entre les agents, ces derniers peuvent échanger des informations indirectement en agissant sur l'environnement commun ou bien directement en s'envoyant des messages de manière souvent asynchrone.

- **L'interaction**

Les agents interagissent au sein de l'environnement dans lequel ils évoluent. Ces interactions ne sont plus réalisées par des invocations de méthodes comme dans le cas des objets mais à travers la communication qui s'effectue généralement de pair à pair. Les interactions permettent ainsi d'exprimer des stratégies de coopération, de collaboration, de compétition, de négociation entre les agents.

- **L'organisation**

Il existe de multiples relations complexes qui unissent les agents et qui peuvent porter sur les buts, les plans, les actions ou les ressources. Ces relations induisent des schémas globaux d'interactions entre les agents. Les organisations permettent donc de formaliser ces schémas et offrent un moyen de spécifier et de concevoir une structure du SMA qui définit l'ensemble des rôles et des relations existant entre ces rôles.

- **L'adaptation**

Un SMA est soumis à divers types de contraintes qui peuvent affecter ses performances. Il est donc essentiel que le système change son comportement lorsqu'il estime qu'il est entrain de dévier de son objectif global ou lorsqu'il s'avère qu'il peut réaliser une meilleure performance. L'adaptation est donc la capacité du système à modifier son comportement en cours de fonctionnement pour l'ajuster dans un milieu dynamique.

- **L'ouverture**

Les SMA sont constitués de plusieurs entités autonomes, hétérogènes, en interaction entre elles au sein d'environnements dynamiques. Au-delà de cette hétérogénéité, les SMA sont caractérisés par l'ouverture qui se manifeste par l'évolution fonctionnelle du

système. Cette évolution correspond à l'ajout, la modification ou la suppression dynamique d'entités du système.

- **L'émergence**

L'émergence est l'apparition progressive de comportements non spécifiés a priori au sein du système. En effet, la fonction globale du système est attendue à partir des spécifications locales de chacun des agents, elle n'est pas programmée à l'avance et elle apparaît comme résultat des interactions des agents entre eux. Cette fonction survient sans organisateur extérieur du système.

- **La situation dans un environnement**

L'environnement d'un SMA est vu comme étant un espace partagé par l'ensemble des agents. C'est le lieu commun au sein duquel les agents agissent et s'influencent les uns les autres. Les agents interagissent avec leur environnement par le biais des perceptions et des actions qu'ils peuvent effectuer sur lui.

- **La délégation**

Dans une application multi-agents, l'utilisateur délègue son contrôle, ou du moins une partie de ce dernier, au système car il ne maîtrise pas le comportement de l'application globale. Cette délégation est due à la complexité de l'application et à l'incapacité de l'utilisateur à gérer toutes les décisions. L'utilisateur délègue son contrôle plus précisément aux agents, dont le caractère autonome et proactif leur permet de prendre les décisions à sa place.

- **La personnalisation**

La personnalisation consiste à ce qu'un agent maintienne les préférences de l'utilisateur en observant ses comportements afin de construire un profil adapté à ce dernier. Ce profil est ensuite utilisé, par exemple, pour aider l'utilisateur à accéder à des informations pertinentes qui le concernent. Cette aide logicielle s'appelle "Profil Utilisateur".

- **L'intelligibilité**

Un système intelligible est un système compréhensible et facilement abordable par les utilisateurs. Dans les SMA, l'intelligibilité découle du caractère anthropomorphique des agents. En effet, on retrouve souvent dans ces systèmes, une abstraction réaliste d'entités du monde réel à travers les agents.

Il est évident que la panoplie des caractéristiques que présentent les SMA fait de ce domaine un axe de recherche très intéressant qui va même devenir un paradigme de programmation à part entière. C'est ce que nous examinons dans le paragraphe suivant.

1.3. La technologie orientée-agent

Face à la complexité croissante des applications informatiques, toujours plus ouvertes, plus hétérogènes et plus dynamiques, la construction de logiciels de haute qualité s'avère une tâche très difficile. En effet, les applications du monde réel sont caractérisées par un nombre important de composants avec des schémas d'interactions en évolution constante et un comportement général soumis aux changements de l'environnement, des services et des utilisateurs. Le rôle de l'ingénierie logicielle consiste donc à fournir les modèles et les techniques adéquats qui permettent de gérer efficacement cette complexité.

Plusieurs paradigmes du génie logiciel, ont tenté d'apporter des solutions à ce problème, mais semblent avoir échoué à deux niveaux essentiels : d'une part, ils manquent de flexibilité dans la manière de définir les interactions entre les différentes entités logicielles, et d'une autre part ils ne fournissent pas suffisamment de moyens pour représenter les structures organisationnelles [Jen00]. Dans cette optique, la technologie orientée-agent est née pour apporter des solutions quant à la capacité de modéliser, concevoir et implémenter des systèmes complexes et distribués et à des niveaux d'abstraction toujours plus élevés.

L'approche orientée-agent a été introduite au début des années 90 par Yoav Shoham [Sho90, Sho93], elle découle principalement du domaine de l'IAD mais s'inspire aussi de domaines très diversifiés tels que le génie logiciel, les systèmes répartis et les sciences sociales. A la lumière des nouvelles notions autour desquelles la technologie agent s'est développée et des nouvelles possibilités de modélisation qu'elle a introduites, elle s'est vue accorder une reconnaissance croissante dans le milieu académique et scientifique et un intérêt grandissant dans le milieu industriel.

La technologie orientée-agent est devenue un paradigme à part entière du génie logiciel disposant de ses propres éléments méthodologiques en termes de conception et de programmation. Les deux dernières décennies ont été marquées par l'apparition d'un très grand nombre de méthodologies et de plates-formes orientées agent. Le nombre d'applications multi-agents développées ne cesse également de s'accroitre et de toucher de plus en plus de domaines critiques avec des enjeux économiques conséquents. C'est

principalement à ces trois derniers éléments que nous nous intéressons dans les paragraphes suivants.

1.4. Méthodologies de conception

Avant de définir une méthodologie orientée-agent, il convient d'abord de disposer de la définition d'une méthodologie en général, Booch [Boo92] la présente comme "un ensemble de méthodes appliquées tout au long du cycle de développement d'un logiciel". Pour Shehory [She01], il s'agit d'un ensemble de guides qui couvrent tout le cycle de vie du développement d'un logiciel en partant de l'analyse des besoins jusqu'au déploiement et la maintenance.

Afin de fournir un cadre propice pour la conception et le développement de systèmes multi-agents, plusieurs méthodologies orientées-agents ont vu le jour depuis le début des années 90. Akbari en recense plus de soixante-quinze apparues entre les années 1991 et 2009 [Akb10]. Toutes ces méthodologies ne s'inspirent pas forcément du même courant de l'ingénierie logicielle. Picard [Pic04] dresse une carte des principales méthodologies ainsi que de leurs sources d'influence comme le montre la figure 1.2 ci-dessous.

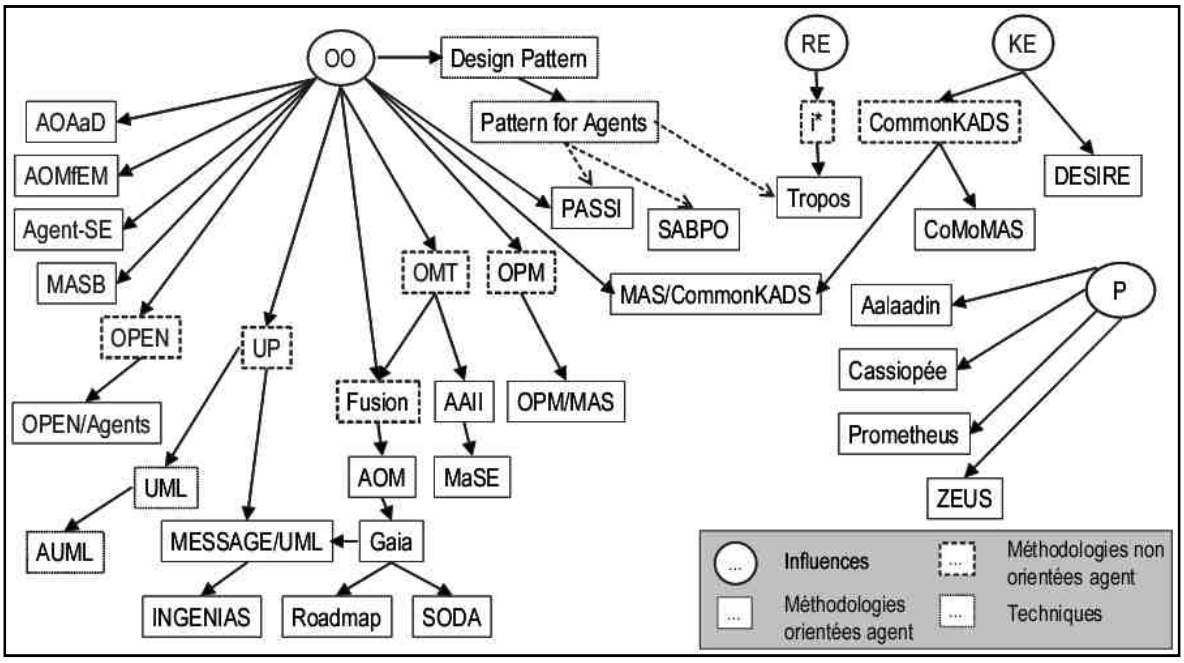

Figure 1.2 : Généalogie des méthodologies orientées-agent [Pic04]

Sans surprises, l'approche orientée-objet (OO) est la principale source d'inspiration pour les méthodologies orientées-agent devant l'ingénierie des connaissances (KE), l'ingénierie

des besoins (RE) et l'approche orientée-plateforme (P). Ainsi la plupart des méthodologies de conception orientées-agent fournissent le cadre nécessaire pour la mise en place de systèmes multi-agents en suivant le processus de développement suivant :

- L'analyse des besoins avec prise en considération des notions de rôle et d'organisation spécifiques au domaine multi-agents.
- La conception de l'architecture avec comme étapes essentielles l'identification des agents qui y interviennent et la définition des interactions entre ces agents.
- Le développement qui correspond aux phases classiques de codage, de test et d'intégration.
- Le déploiement et la maintenance qui correspondent à la correction d'éventuelles erreurs ou l'ajout de fonctionnalités demandées par l'utilisateur.

Les méthodologies orientées-agent existantes peuvent ou non être couplées à des plates-formes d'exécution particulières. C'est précisément à quoi nous nous intéressons dans le paragraphe suivant.

1.5. Plates-formes de développement

A l'image des méthodologies de conception, ces dernières années ont vu se multiplier les plates-formes orientées-agent dont le but est de permettre la mise en œuvre de modèles à base d'agents. En effet, les outils et logiciels issus de la technologie orientée-objet ne sont pas suffisamment adaptés au concept d'agent dans la mesure où ils ne permettent pas d'exprimer les caractéristiques d'autonomie, de proactivité, de réactivité des agents ainsi que les caractéristiques de socialité et de dynamique du SMA. Des plates-formes et outils de développement orientés-agents ont donc vu le jour pour offrir un support de mise en œuvre adéquat aux développeurs de SMA. Certains se basent sur des formalismes agent spécifiques et d'autres sont issus de la continuité de l'approche objet.

Arlabosse et al. [Arl04] classent les plates-formes orientées-agent en trois catégories :

- Les plates-formes de simulation : elles sont particulièrement adaptées aux problèmes de simulation qui consistent à reproduire le comportement d'un système ou d'un phénomène afin d'en étudier la complexité.
- Les plates-formes d'exécution : elles fournissent des outils d'implémentation pour des modèles multi-agents variés et ne sont pas associées à des méthodologies de conception particulières.

- Les plates-formes de développement : elles offrent un support complet au développement de systèmes multi-agents suivant une méthodologie prédéfinie. Ce support intègre toutes les notions SMA et fournit tous les outils nécessaires pour leur mise en œuvre.

1.6. Domaines d'application

Les SMA sont un cadre propice pour la conception et la réalisation de systèmes complexes, ouverts, coopératifs, décentralisés et à des niveaux d'abstraction de plus en plus évolués. A l'heure actuelle, ils possèdent des champs d'application très vastes couvrant des domaines scientifiques très variés. L'utilisation des SMA s'avère particulièrement utile dans les domaines où l'aspect de décentralisation est important tels que les réseaux informatiques ou de télécommunications. La flexibilité et la richesse des interactions ou encore la mobilité sont des notions de l'approche agent dont l'exploitation est très intéressante dans les réseaux quelque soit leur nature (fixes, mobiles ou ambiants). Le web présente aussi un terrain propice à l'utilisation des agents intelligents notamment en ce qui concerne la recherche d'informations et le commerce électronique. Un autre domaine où les SMA ont fait preuve d'efficacité concerne les applications industrielles telles que l'automatisation et le contrôle de processus, la production, la gestion du transport et du trafic aérien, la robotique, etc. Par ailleurs, les SMA apportent une solution puissante au concept de simulation en permettant de modéliser directement des individus, leurs comportements et les relations qui les unissent. Ils fournissent un outil expérimental qui permet de simuler le déroulement de divers processus sociaux et d'éclaircir l'émergence de certains comportements aussi bien chez les sociétés humaines que chez les sociétés animales.

Selon Boissier et al. [Boi04] les champs d'application des SMA peuvent être regroupés en trois grandes catégories :

- **La résolution :** Cette catégorie englobe les applications destinées résoudre des problèmes complexes tels qu'ils sont définis en IA classique, étendue à un contexte d'informatique distribuée. Dans ce cadre, les agents participent à la résolution d'un problème global en se partageant les tâches, les ressources et les connaissances.

- **L'intégration :** Cette catégorie englobe les applications qui visent à intégrer des logiciels existants, des systèmes mécaniques et des êtres humains et à faire interopérer l'ensemble d'une manière cohérente. On peut citer par exemple les applications de commerce électronique, de collecticiel ou d'informatique diffuse.

- **La simulation :** Cette catégorie englobe les applications destinées à modéliser et à reproduire des phénomènes du monde réel tels que les phénomènes sociaux, environnementaux ou éthologiques, afin de les comprendre et d'expliquer les comportements émergents.

2. Évaluation des systèmes multi-agents

L'apparition des SMA et des premiers travaux s'inscrivant dans le cadre de ce domaine de recherche, s'est progressivement accompagnée d'un certain nombre d'analyses et de travaux d'évaluation portant, dans un premier temps, sur la technologie agent elle-même, et plus tard sur les méthodologies de conception et les outils de développement des SMA.

2.1. Évaluation de la technologie orientée-agent

La plupart des travaux s'étant penchés sur l'évaluation de la technologie orientée-agent s'intéressent à deux aspects essentiels : d'une part la capacité de cette approche à gérer efficacement la complexité des problèmes du monde réel, et d'une autre part sa valeur ajoutée par rapport aux approches existantes, notamment la technologie orientée-objet.

Ainsi, dans le but d'examiner le premier aspect, dans [Jen00], Jennings évalue l'adéquation entre les concepts essentiels apportés par le paradigme agent d'une part et les techniques clés permettant de gérer la complexité des problèmes du monde réel d'une autre part (voir Tableau 1.1 ci-après).

Ces techniques sont :

- La décomposition qui consiste à diviser le problème initial en plusieurs sous-problèmes plus faciles à résoudre.
- L'abstraction qui consiste à définir un modèle simplifié du système étudié tout en considérant certaines propriétés et en négligeant d'autres, estimées moins importantes.
- L'organisation qui consiste à identifier et gérer les interrelations entre les diverses composantes du système.

Le tableau suivant (Tableau 1.1) résume les concepts clés de la technologie orientée-agent et la manière dont ils permettent d'aborder les problèmes liés à la représentation et la gestion des systèmes complexes.

Problèmes des systèmes complexes	Technique	Concepts multi-agents
– Distribution des composants – Complexité du contrôle – Absence de contrôle global – Imprévisibilité des interactions	Décomposition	– Distribution des tâches – Autonomie des agents – Décentralisation du contrôle – Flexibilité des interactions
– Modélisation des sous-systèmes et de leurs composantes – Modélisation des interactions – Expression des relations organisationnelles	Abstraction	– Organisations d'agents – Capacités sociales (coopération, coordination, négociation…) – Gestion des organisations (rôles, normes, lois sociales…) et de leurs structures (groupes, équipes…)
– Structures complexes – Relations organisationnelles instables	Organisation	– Capacité de formation, maintien et dissolution des organisations de manière flexible

Tableau 1.1 : Problèmes des systèmes complexes et solutions de la technologie agent

Cette analyse établit que la décomposition en agents semble être un moyen efficace de partitionner un problème complexe, et que les abstractions clés du domaine agent offrent un cadre propice à la modélisation de tels systèmes. La philosophie de l'approche agent dans l'identification et la gestion des relations organisationnelles permet de bien aborder les dépendances et les interactions qui y existent [Jen00].

Le second aspect selon lequel la technologie orientée-agent a été évaluée est son apport en comparaison aux technologies existantes, en particulier l'orientée-objet.

En effet, il existe un nombre important de similarités entre l'approche orientée-agent et l'approche orientée-objet. Il s'agit d'une conséquence au fait que la notion d'agent repose sur la notion d'objet dont elle découle et qu'elle étend [Sho93]. Toutefois, les différences entre ces deux approches sont encore plus importantes.

Dans [Woo00], Wooldridge et Ciancarini comparent les agents et les objets afin de souligner l'apport de la technologie agent au génie logiciel, toujours dans le but de modéliser, concevoir et mettre en œuvre des systèmes complexes. Dans ce contexte, au moins trois principales différences entre les objets et les agents ont été soulevées, à savoir :

- L'autonomie : les agents sont des entités qui possèdent le contrôle de leur comportement dans le sens ou elles décident pour elles-mêmes d'effectuer ou non une action suite à la demande d'un autre agent. Les objets, au contraire, ne possèdent pas le choix des actions qu'ils effectuent et répondent systématiquement aux invocations de leurs comportements publics de la part des autres objets.
- La flexibilité : qui se traduit par la réactivité, la proactivité et la capacité sociale des agents. Ces types de comportements ne sont pas du tout définis pour les objets qui ne disposent que de schémas d'interaction prédéfinis et rigides.
- Le comportement actif : les agents sont des entités actives engagées indéfiniment dans une boucle perception-action, contrairement aux objets qui sont des entités passives pouvant s'activer ponctuellement suite à l'invocation d'une de leurs méthodes.

Outre ces différences, Tveit [Tve01] souligne la capacité des agents à modéliser un système donné à un haut niveau d'abstraction. En effet, les agents ont en plus, par rapport aux objets, la capacité de représenter des notions mentales telles que les croyances, les intentions, etc. et peuvent s'engager dans des interactions de haut niveau en se basant sur la théorie des actes du langage, contrairement aux objets qui échangent des messages de manière ad-hoc.

Ainsi, dans l'approche orientée-agent, un système n'est plus considéré comme une collection d'entités passives comme il l'est dans le paradigme objet, mais plutôt comme un ensemble d'entités actives, capables d'agir de manière autonome, ayant le contrôle de leurs propres comportements, et pouvant interagir et s'organiser de manière flexible [Lin00]. Cette vision des systèmes complexes est différente des approches traditionnelles dans le sens où elle admet que le concepteur n'est plus tenu de spécifier, à priori et dans ses moindres détails, la dynamique du système. Au contraire, dans une approche agent, le concepteur ne fait que définir l'état initial du système et spécifier les buts locaux des différents agents qui le composent. Par conséquent, et en absence de contrôle central, le comportement et la dynamique globale du système sont les résultats des interactions entre ses entités. Ainsi, la technologie orientée-agent est beaucoup plus adaptée au développement des systèmes complexes à comportement émergeant, dans la mesure où elle permet de réduire la difficulté de la tâche du concepteur relative à l'anticipation des interactions entre les différentes entités du système [Pet01].

En conclusion, nous remarquons que tous les travaux recensés dans la littérature sur l'évaluation de la technologie orientée-agent abordent cette dernière de manière qualitative. Il aurait été plus pratique si l'on disposait de données quantitatives qui montrent, en se basant sur un ensemble standard de métriques logicielles, l'apport de l'approche agent par apport aux autres approches notamment en termes de productivité, de fiabilité, de maintenance, etc [Jen00]. Le problème est que de telles données n'existent pas et par conséquent les arguments présentés en faveur de l'approche orientée-agent étaient purement qualitatifs.

2.2. Évaluation des méthodologies de conception

L'apparition du paradigme agent date du début des années 90 [Sho90, Sho93]. Les années suivantes ont été marquées par l'introduction d'un nombre considérable de méthodologies orientées-agent dans le but d'assister et de guider les concepteurs d'applications et de systèmes à base d'agents. Comme déjà cité auparavant, Akbari recense plus de 75 méthodologies apparues entre les années 1991 et 2009 [Akb10]. Ceci a suscité un certain nombre d'interrogations, notamment : à quel point les méthodologies existantes répondent-elles aux besoins des programmeurs ? Dans quelle mesure sont-elles adaptées au développement des systèmes multi-agents ? Dans ce cas, quelle méthodologie adopter et pour quelles raisons ?

Afin de répondre à ces questions, plusieurs travaux sur l'évaluation des méthodologies orientées-agent ont vu le jour. Shehory et Sturm sont les premiers à s'intéresser à cette problématique [She01]. Ils définissent plusieurs critères d'évaluation scindés en deux catégories :

- des critères qui relèvent du domaine du génie logiciel en général et qui permettent d'évaluer la qualité d'une méthodologie en estimant à quel point elle répond aux exigences suivantes : la précision, l'accessibilité, l'expressivité, la modularité, la gestion de la complexité, l'exécutabilité, l'ouverture, etc.
- des critères qui relèvent du domaine de l'orienté-agent reflétant la capacité de la méthodologie à représenter les notions d'autonomie, de complexité, d'adaptation, de parallélisme, de distribution et de richesse de la communication.

Chacun des critères ainsi définis est estimé qualitativement pour chaque méthodologie examinée, il lui est attribué un qualificatif parmi les suivants : bon (+), satisfaisant (*), insatisfaisant(-) et non supporté (NS), permettant d'indiquer à quel point le critère est assuré. L'approche d'évaluation proposée est utilisée pour comparer trois méthodologies

orientées-agent qui sont : GAIA, ADEPT et DESIRE. Cette même approche est reprise plus tard par Juneidi et Vouros dans [Jun04] pour évaluer et comparer les méthodologies GAIA, MaSE et AUML.

Cernuzzi et Rossi [Cer02] définissent une approche d'évaluation utilisant une métrique et des valeurs quantitatives. Il s'agit d'une application de l'approche des métriques logicielles Goal-Question-Metric (QGM). Elle se décline en plusieurs étapes dont la plus importante est la définition d'un arbre d'attributs mesurables. Il s'agit précisément de définir des critères d'évaluation, de les classifier et de spécifier un arbre en identifiant les critères généraux et en les spécialisant jusqu'à obtenir des critères plus fins et quantifiables. Ainsi, il est possible de mesurer tous les critères en partant des feuilles jusqu'à la racine de l'arbre.

Cernuzzi et Rossi [Cer02] définissent dans ce cadre trois grandes classes de critères d'évaluation :

- Critères relatifs à des attributs internes aux agents tels que l'autonomie, la réactivité, la proactivité et les notions mentales ;
- Critères relatifs à des attributs d'interaction tels que la capacité sociale, l'interaction avec l'environnement, etc. ;
- Critères relatifs à des besoins non fonctionnels tels que la modularité, l'abstraction, le support de communication, etc.

Une autre classification des critères d'évaluation des méthodologies orientées-agent est proposée par Sabas et al. [Sab02] dans le cadre d'un framework labellisé MUCCMAS (MUltidimensional framework of Criteria for the Comparison of MAS methodologies). Ce dernier se base sur six dimensions, chacune englobant un certain nombre de critères :

- La dimension méthodologie qui couvre sept critères à évaluer, notamment les étapes du processus, les modèles et l'approche de développement, le degré et le moment d'implication de l'utilisateur, la réutilisabilité des modèles et la disponibilité du support logiciel et méthodologique.
- La dimension représentation qui contient quatre critères à savoir le découpage du système, le formalisme, le séquençage et la qualité des modèles
- La dimension agent, qui représente l'apport principal de ce travail, s'intéresse aux caractéristiques propres des agents tels que leurs natures (homogènes ou

hétérogènes), leurs types (intelligents, mobiles, etc.) et leurs attributs (autonomie, adaptation, coopération, communication, etc.)

- La dimension organisation qui couvre l'aspect structurel du système. Les critères qui y sont associés sont l'image de l'organisation (hiérarchique, distribuée, holonique, etc.), la nature de l'environnement (structuré, stable, observable, etc.), son type (actif ou passif) et la nature des données utilisées (numériques ou symboliques).
- La dimension coopération qui s'intéresse aux modes et langages de la communication, au modèle d'interaction et au type du contrôle.
- La dimension technologie qui examine les caractéristiques du logiciel visé tels que le mode de traitement, le type de programmation, la nature de l'interface homme-machine et l'environnement de développement.

L'approche proposée a été utilisée pour évaluer et comparer neuf méthodologies orientées-agent, l'évaluation réalisée est purement qualitative, aucune mesure n'a été proposée pour ce faire.

Selon Sturm et Shehory [Stu03], les travaux existants sur l'évaluation des méthodologies orientées-agent ne couvrent pas de manière exhaustive tous les concepts constituant une méthodologie complète. Afin de pallier cette limitation, Sturm et Shehory proposent un framework d'évaluation qui aborde les quatre majeures facettes d'une méthodologie qui sont les suivantes :

- Concepts et propriétés : cette facette correspond aux caractéristiques intrinsèques des agents et des SMA telles que l'autonomie, la réactivité, la proactivité, la socialité, les croyances, les désirs, les intentions, etc.
- Notations et techniques de modélisation : cette facette s'intéresse aux propriétés auxquelles le langage de modélisation doit répondre, notamment la facilité d'utilisation, la précision, l'expressivité, la gestion de la complexité, etc.
- Processus de développement : cette facette est évaluée selon deux critères, le contexte de développement et la couverture du cycle de vie.
- Pragmatique : cette dernière facette concerne les aspects pratiques du déploiement et d'utilisation d'une méthodologie, elle est évaluée selon plusieurs critères dont la disponibilité des ressources, le besoin d'expertise, l'adéquation à un langage, une architecture et un domaine d'application et enfin le passage à l'échelle.

L'évaluation de ces critères est abordée de manière qualitative. A chaque critère examiné est assignée une valeur sur une échelle de 1 à 7. La valeur 1 indique que la méthodologie ne couvre pas du tout le critère alors que la valeur 7 indique que la méthodologie couvre le critère de manière complète. L'approche proposée a été utilisée pour analyser la méthodologie GAIA. Une extension de cette approche a été proposée par Lin et al. [Lin07] pour évaluer et comparer les méthodologies TROPOS, GAIA et MaSE.

Les travaux d'Elamy et Far [Ela08] se démarquent de leurs prédécesseurs par le fait que l'approche d'évaluation proposée se base sur une étude statistique. En effet, les auteurs proposent un ensemble de critères d'évaluation organisés sous la forme d'un arbre et labellisé Multidimensional Weighted-Attributes Framework (MWAF). L'idée principale consiste à assigner à chaque critère d'évaluation un poids qui exprime son importance et une valeur numérique qui exprime son degré de représentation.

En conclusion, on ne peut que souligner l'abondance, aussi bien des méthodologies de conception orientées-agent que des travaux d'évaluation qui s'y penchent. Nous avons présenté les travaux les plus importants dans ce contexte en mettant le point sur les critères d'évaluation adoptés ainsi que sur la manière dont ces derniers ont été estimés. Les résultats de l'application de ces différentes méthodes pour l'évaluation et la comparaison des méthodologies orientées-agent existantes n'ont pas été exposés car ils sont sans intérêt manifeste pour nos travaux.

2.3. Évaluation des plates-formes de développement

D'après Sudeikat, l'évaluation des méthodologies de conception orientées-agent n'est pas suffisante en elle-même, elle doit absolument être complétée par la considération des plates-formes de développement [Sud04]. En effet, face à la multitude des plates-formes existantes, les développeurs de systèmes multi-agents ont besoin d'un moyen leur permettant de choisir celle qui correspond au mieux à leurs exigences et aux objectifs de leurs projets. C'est dans ce contexte, que nous nous intéressons ici aux travaux d'évaluation des plates-formes orientées-agent.

Dans cette perspective, Ricordel et al. [Ric00] définissent quatre étapes fondamentales dans l'ingénierie des systèmes multi-agents à savoir, l'analyse, la conception, le développement et le déploiement. Ils proposent ensuite une approche d'évaluation qui fait référence à ces quatre étapes. Les critères utilisés sont : la complétude, l'applicabilité, la complexité et la réutilisation. Ces derniers sont caractérisés de manière purement

qualitative et ont permis d'évaluer et de comparer quatre plates-formes orientées-agent qui sont AgentBuilber, Jack, Madkit et Zeus.

Boissier et al. [Boi02] présentent aussi plusieurs critères utiles pour évaluer et comparer les plates-formes orientés-agent. Dans le but d'estimer dans quelle mesure les plates-formes évaluées prennent en considération les différents aspects du processus de développement logiciel, les critères définis ont été groupés en cinq catégories présentées comme suit :

- Caractéristiques générales : cette catégorie définit des critères généraux tels que la nature de la plate-forme et son état, le nombre de ses utilisateurs, le nombre d'applications développées, le type d'application privilégié, etc.
- Modèles multi-agents : cette catégorie regroupe des critères centrés sur les concepts multi-agents disponibles sur la plate-forme tels que les caractéristiques individuelles et sociales des agents, la nature des organisations et leurs structures, le type de communication, son modèle et sa structure, le modèle de l'environnement, etc.
- Caractéristiques physiques : dans cette catégorie, les critères présentés portent sur des aspects physiques tels que la nature de l'implémentation des agents, la nature du moteur d'interaction, la structure de l'environnement, le type de distribution physique, la tolérance aux pannes, etc.
- Environnement de développement : cette catégorie définit des critères relatifs au processus de développement tels que la méthodologie d'analyse, la disponibilité du langage de description des agents, l'existence de bibliothèques et la disponibilité de la documentation.
- Environnement d'exécution : cette catégorie regroupe des critères centrés sur l'utilisation de la plate-forme tels que la disponibilité d'outils d'administration, de monitoring et mise au point, les places du concepteur et de l'utilisateur et le support logistique.

L'utilisation de certains de ces critères nécessite une quantification, cependant, aucune méthode de mesure n'a été proposée pour estimer ces derniers et l'approche proposée n'a pas été testée pour l'évaluation et la comparaison de plates-formes orientées-agents.

Dans [Ngu02], un autre ensemble de critères est proposé pour l'évaluation des plates-formes multi-agents tels que la compatibilité aux standards, la communication, la mobilité des agents, la sécurité, la disponibilité, l'utilisation et le développement.

Garneau et al. [Gar02] évaluent et comparent huit plates-formes orientées-agent choisies en raison de leur popularité et de leur accessibilité. L'analyse se base sur quinze critères dont la facilité d'utilisation, la facilité de transition entre le la conception et l'implémentation, la souplesse de l'outil, la simplicité de mise en œuvre, l'extensibilité du code, etc. Dans ce travail, les auteurs estiment chaque critère en lui attribuant une valeur de 0 à 4 en fonction du degré de prise en charge de ce dernier par la plate-forme.

Dans [Les04], l'étude se concentre uniquement sur les plates-formes conformes aux spécifications FIPA. Neuf plates-formes ont été évaluées et comparées selon des critères généraux tels que la documentation, la mise à jour, la popularité, l'accessibilité, etc. Les critères proposés ne sont pas spécifiques aux environnements agent, c'est pour cette raison que cette étude reste très générale. Des critères plus spécifiques ont été proposés dans [Sha04] pour évaluer et comparer les plates-formes JADE, Jack et Zeus. L'étude s'intéresse particulièrement à la performance du système de transport de messages en se basant sur le temps moyen d'aller-retour des messages.

Mulet et al. [Mul06] proposent de comparer les plates-formes multi-agents en évaluant deux de leurs services usuels, il s'agit du service de messagerie et du service d'annuaire. Pour le premier, le critère adopté est la valeur du temps d'aller-retour (RTT), il est mesuré en faisant varier le nombre de paires d'agents communicants et le nombre de messages échangé entre chaque paire. Pour le second, les critères adoptés sont le temps d'enregistrement d'un agent et le temps de recherche d'un service dans l'annuaire. Ces critères sont mesurés en faisant varier le nombre d'agents déjà enregistrés dans l'annuaire. Les plates-formes comparées ont été sélectionnées sur la base de la disponibilité de leurs codes sources et pour leur conformité au standard de la FIPA.

Dans [Tri07], l'évaluation de concentre sur les plates-formes pour agents mobiles. Les auteurs procèdent à une évaluation qualitative en comparant plusieurs propriétés de ce type de plates-formes. L'étude est complétée par des expérimentations afin de déterminer la plate-forme qui présente le meilleur comportement face à divers scénarios de test.

En conclusion, nous notons qu'à l'instar des travaux sur l'évaluation de l'approche agent et de ses méthodologies de conception, les travaux sur l'évaluation des plates-formes sont nombreux. Ce qui n'est pas le cas des modèles et des applications multi-agents développés. C'est ce que nous proposons d'étudier à travers les paragraphes suivants.

2.4. Validation de modèles multi-agents

La validation de modèles à base d'agents s'inscrit dans le cadre général de la validation des modèles de simulation destinés à décrire, expliquer et prédire des phénomènes du monde réel. Elle concerne donc particulièrement les SMA construits dans un objectif de simulation. Il s'agit d'une étape importante et indispensable pour construire des modèles multi-agents fiables et efficaces [Amb06].

Dans ce cadre, il est rappelé dans [Man03] que la simulation d'un système du monde réel commence par l'élaboration d'un modèle conceptuel qui sera ensuite instanciée grâce à des données particulières pour construire un modèle logiciel (dans notre cas le modèle multi-agents). Ce dernier est ensuite exécuté et génère des données résultats. En adoptant cette vue, la validation des modèles multi-agents se décline en trois types comme le montre la figure suivante [Lou08] :

- La validation du modèle conceptuel : elle consiste à évaluer à quel point ce modèle permet de représenter le système ou le phénomène du monde réel ;
- La vérification du modèle logiciel : elle vise à déterminer dans quelle mesure le modèle multi-agents développé est conforme à sa conception.
- La validation opérationnelle : elle permet de mesurer le degré d'adéquation entre les données générées par le modèle multi-agents et les données du phénomène réel qu'il est supposé représenter.

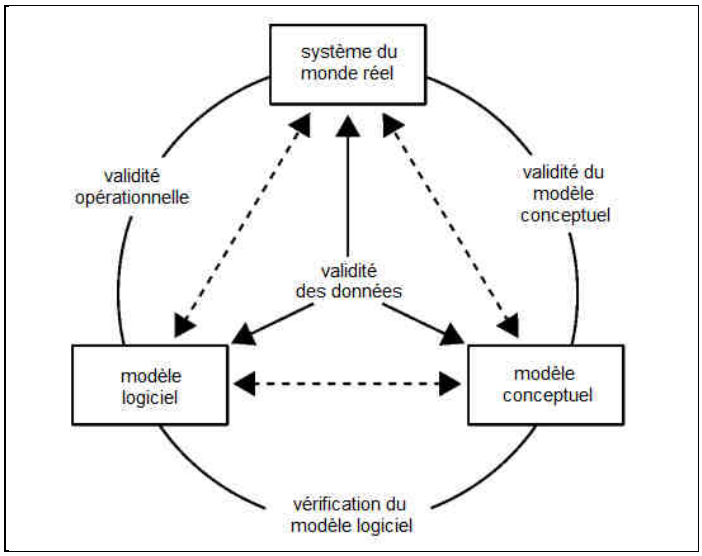

Figure 1.3 : Représentation des composantes de la validation des modèles [Lou08]

2.5. Évaluation des applications multi-agents

Ce type d'évaluation auquel nous nous intéressons concerne les applications multi-agents implémentées et se concentre particulièrement sur les performances du SMA indépendamment de la méthode de conception et de l'outil de développement utilisés.

Commençons par définir l'évaluation des performances en général ainsi que ses objectifs et techniques. Il s'agit d'une tâche très importante et la plus critique de la construction de n'importe quel système informatique [Wil05]. Elle consiste à déterminer les différents aspects de la performance d'un système et de les estimer quantitativement ou qualitativement. Il n'existe pas de définition générique d'une mesure de performance, elle dépend du système et sa détermination nécessite la bonne compréhension de ce dernier et de son utilisation [Leb10]. Selon Jain [Jai91] il s'agit d'un « art du génie logiciel ».

Les objectifs de l'évaluation des performances sont multiples, elle peut servir à dimensionner le système évalué, à comparer différents systèmes, ou bien à détecter d'éventuels problèmes tels que les goulets d'étranglement.

Il y a trois techniques de base pour l'évaluation des performances [Jai91] :

- La modélisation analytique qui consiste à représenter le système à évaluer par un modèle mathématique abstrait. Il s'agit de trouver un formalisme qui permette de traduire le comportement et décrire certains aspects du système. Le modèle ainsi établi est analysé numériquement dans le but de dégager les paramètres de performances du système réel.
- La simulation qui consiste à implémenter un modèle logiciel qui imite de manière simplifiée le comportement du système réel à évaluer. Une simulation est donc une reproduction de l'évolution du système pas à pas, le temps y étant représenté plus lentement que le temps réel. Il s'agit de la technique la plus utilisée car elle est relativement simple par rapport à la modélisation analytique et permet de mettre en évidence les aspects les plus importants de la performance d'un système.
- La technique de mesures expérimentales qui consiste à mesurer directement certaines caractéristiques du système réel et d'analyser les résultats de ces mesures. Il s'agit de munir le système à évaluer d'instruments spécifiques (sondes) qui permettent de capturer les valeurs nécessaires à l'évaluation et de les sauvegarder, un moniteur se charge alors de collecter ces valeurs, de mesurer la performance et de présenter le résultat à l'utilisateur.

L'évaluation des performances est largement abordée dans les systèmes informatiques en général et dans les systèmes répartis en particulier. Cependant, dans les SMA il lui a été réservé peu d'intérêt. Les travaux réalisés dans ce contexte n'abordent pas cette question d'un point de vue générique et présentent des solutions souvent liées au domaine d'application ou aux données du problème initial. La revue de la littérature liée à cette thématique de recherche a permis de dégager deux catégories d'évaluations : l'évaluation de l'adéquation fonctionnelle ou de l'utilité du système et l'évaluation des performances globales.

2.5.1. Évaluation de l'adéquation fonctionnelle

L'adéquation fonctionnelle est définie dans [Kad09] comme étant l'utilité du système. Elle peut être évaluée par une entité externe à ce dernier. Ceci est réalisé en calculant une fonction d'utilité globale qui permet d'exprimer dans quelle mesure le SMA réalise les objectifs pour lesquels il a été conçu. Dans ce contexte, la fonction d'utilité est définie à priori et les paramètres dont elle dépend sont étroitement liées au problème à résoudre [Tho07].

L'évaluation de l'adéquation fonctionnelle peut également être effectuée par le SMA lui-même, dans le cadre de la résolution d'un problème, afin de déterminer s'il a atteint ou non une solution optimale. Une mesure possible pour évaluer ce critère est la distance entre la solution obtenue et la solution optimale [Kad09]. Dans ce contexte, Gnanasambandam et al. proposent dans [Gna05] une méthodologie pour contrôler la performance d'un réseau d'agents distribués. Le SMA est doté d'une capacité d'auto-évaluation, ainsi il lui revient de prédire sa performance en calculant sa fonction d'utilité. Cette dernière est définie en utilisant des métriques telles que le délai d'acheminement de bout en bout et la latence.

Le problème majeur avec cette méthode d'évaluation c'est qu'elle ne peut pas être facilement généralisée. En effet, il n'y a pas de définition générale d'une fonction d'utilité, cette dernière est toujours dépendante du système et est étroitement liée au problème à résoudre. Bouzouita [Bou13] se penche sur l'étude de ce problème et tente de définir une approche générique pour l'évaluation des performances d'un SMA en s'appuyant sur la notion d'utilité. L'approche proposée permet d'abord d'évaluer le degré d'adéquation globale du système. Elle permet ensuite de mesurer la qualité du résultat obtenu en se basant sur des degrés de satisfaction locale des agents et sur le degré de satisfaction global de tout le SMA.

2.5.2. Évaluation des performances

Malgré le grand besoin de disposer de moyens d'évaluation des performances globales des SMA, peu de travaux se sont intéressés à cette problématique. En effet, la plupart des travaux existants proposent des solutions dépendantes du domaine d'application ou bien spécifiques à un problème particulier. Par exemple dans [Oma00] les auteurs effectuent une comparaison entre les SMA mobiles et statiques. Pour ce faire, les performances de deux SMA de recherche de texte ont été évaluées en se basant sur un critère unique qui est le temps d'exécution d'une recherche.

Hu B. [Hub04] examine la manière dont les comportements locaux des agents peuvent affecter les performances globales du SMA. Les cas d'étude concernent RoboNBA qui est une plateforme de jeu de basketball pour robots autonomes. Les comportements locaux étudiés sont très spécifiques au domaine d'application. Il s'agit notamment des stratégies de passage du ballon, les stratégies d'attaque ou de défense, etc. Pour les performances globales, il s'agit également de critères spécifiques tels que le temps de contrôle du ballon, la précision du passage du ballon, etc.

Le travail décrit dans [Bab05] présente une analyse des performances et une comparaison de deux SMA de recherche d'information. Le premier se compose uniquement d'agents stationnaires, et le second comporte un agent mobile. L'évaluation est réalisée moyennant la technique de simulation et les mesures de performance sont principalement la probabilité et le temps moyen de réception de la bonne information, qui sont des critères très spécifiques.

Dans [Zöl06], les auteurs évaluent les performances d'un SMA destiné à la planification des admissions des patients dans un hôpital. L'évaluation se base particulièrement sur des critères spécifiques tels que le temps d'attente des patients et le taux d'occupation des différentes ressources de l'hôpital. Toutefois, un critère plus général est adopté, il s'agit du temps de négociation des agents pour la satisfaction des demandes en temps réel.

Joumaa et al. [Jou08] s'intéressent à l'évaluation des interactions dans un SMA simulant une société de robots. L'analyse porte sur les interactions dites « pertinentes » dans le sens où elles permettent de modifier les états internes des agents. Le poids de chaque interaction est ainsi mesuré en se basant sur cette définition.

Dans [Lee98], la performance d'un SMA est définie en termes d'indicateurs classiques tels que le débit, le temps de réponse, le temps d'exécution et la charge de communication. Des indicateurs de même nature sont définis dans [Bon08], nous citons : la consommation en ressources, le temps de réponse, le nombre de tâches réalisées, le temps de calcul et la charge de communication.

Enfin, Kaddoum et al. [Kad09] s'intéressent à l'évaluation des SMA auto-adaptatifs. Pour ce faire, ils proposent des critères classés en trois catégories :

- Des critères relatifs à l'exécution du système tels que le temps, la charge de communication, la précision et la qualité de la solution et l'utilisation de la mémoire.
- Des critères relatifs aux propriétés intrinsèques du système tels que la complexité algorithmique de son comportement et la décentralisation.
- Des critères relatifs à la méthodologie tels que la généricité, la distribution et la facilité de déploiement.

L'étude est intéressante dans son ensemble car elle permet de couvrir divers aspects des SMA. Néanmoins les critères proposés sont analysés et interprétés de manière étroitement liée à la propriété d'auto-adaptation. De plus, aucun moyen de mesure de ces critères n'a été proposé ni testé sur un exemple concret.

Conclusion

A travers ce chapitre, nous avons présenté l'état de l'art se rapportant à la question de l'évaluation dans les systèmes multi-agents. Nous avons pu à travers cette étude dégager le bilan de constats suivants :

- Les SMA possèdent des caractéristiques fonctionnelles propres qui font leurs différences avec les systèmes informatiques classiques et qui rendent leur analyse plus ardue. Les approches d'évaluation classiques ne leur sont donc pas suffisantes.
- Les travaux existants sur l'évaluation des SMA se concentrent essentiellement sur les cadres conceptuels et exécutifs. La plupart des travaux s'intéressent à l'évaluation des méthodologies de conception et des plates-formes d'exécution, bien que peu d'autres travaux traitent la validation et la vérification de modèles multi-agents.
- Les travaux d'évaluation des applications multi-agents indépendamment de la méthodologie et de la plate-forme sont peu nombreux.

– Les solutions existantes pour l'évaluation des applications multi-agents sont très liées au domaine d'application ou au problème traité par le SMA en question et ne peuvent donc pas être généralisées.

Au vu de tous ces constats, nous nous proposons de nous intéresser dans cette thèse à l'évaluation des applications multi-agents sans tenir compte de la méthodologie adoptée ou de la plate-forme utilisée et tout en respectant les spécificités des SMA dans notre démarche. Dans le reste de ce manuscrit, nous détaillons notre approche de résolution de cette problématique.

CHAPITRE 2

UNE ARCHITECTURE D'ÉVALUATION DES SMA BASÉE SUR L'OBSERVATION

Sommaire

Au vu des limites que présentent les travaux d'évaluation existants dans le domaine des SMA, nous nous proposons dans ce chapitre de définir une solution pour l'évaluation des applications multi-agents. Nous commençons par décrire notre approche de résolution ainsi que l'architecture globale du système d'évaluation proposé. Ce dernier se base essentiellement sur la notion d'observation laquelle est présentée ensuite. La solution mise en avant pour l'observation des SMA est enfin détaillée.

1. Approche d'évaluation et architecture proposée

Selon Boissier et al. [Boi04], le développement d'applications multi-agents, comme tout développement de logiciels, met en évidence un ensemble de besoins fonctionnels auxquels ces dernières peuvent répondre grâce à leurs caractéristiques énumérées dans la section 1.2 du chapitre 1. Ces caractéristiques, qu'elles soient à l'origine d'un choix de conception initial ou qu'elles s'en suivent, permettent de réaliser un certain nombre de besoins non fonctionnels tels que la performance, la robustesse, la fiabilité, la portabilité, l'extensibilité, etc.

Il est utile de mentionner ici que les caractéristiques fonctionnelles que possèdent les SMA leurs sont propres et permettent de les distinguer par rapport aux systèmes distribués classiques. Ainsi, les approches d'évaluation existantes, ne tenant pas compte des propriétés spécifiques des SMA, ne permettent pas de les évaluer efficacement. L'analyse des performances d'un SMA ne peut pas se limiter à des critères généraux et usuels tels que le temps d'exécution, l'utilisation de la mémoire et la charge de communication, ni même à des critères spécifiques tels que à la qualité et la précision de la solution qu'il permet d'obtenir. La performance d'un SMA est une notion plus élaborée, plus complexe à l'image de ce dernier et devrait être évaluée grâce à des indicateurs découlant des propriétés intrinsèques de tels systèmes. C'est dans cette optique, que nous proposons une approche d'évaluation des performances des SMA par caractérisation de leurs propriétés fonctionnelles c'est-à-dire en procédant à leur analyse détaillée et description précise.

Cependant, la première difficulté à laquelle nous sommes confrontés consiste à aller au-delà des frontières d'un SMA, à ne plus le considérer comme une boîte noire et trouver le moyen adéquat d'appréhender sa structure et son comportement en vue de son évaluation. La tâche s'avère d'autant plus délicate que les caractères d'autonomie des agents, d'asynchronisme de leurs interactions et de parallélisme de leurs traitements, en plus de la décentralisation du contrôle, rendent impossible la prédiction de l'état global

du système à un instant donné. De même, ces caractères favorisent l'émergence éventuelle de structures et/ou de comportements non spécifiés a priori au sein du SMA et rendent son analyse plus difficile.

Par conséquent, la mise en place d'une solution appropriée pour l'évaluation d'un SMA nécessite la connaissance précise de son état à tout moment de son exécution. L'acquisition de cette connaissance ne peut être assurée que si l'on dispose d'un moyen efficace pour observer le SMA en question à un niveau de granularité adéquat. L'observation de l'exécution d'un SMA s'avère donc une phase préalable indispensable pour l'évaluation des sa performance dans la mesure où elle permet de fournir les informations nécessaires à cet effet. Ces informations constituent la base à partir de laquelle le système d'évaluation calcule les mesures de performance. Ainsi, la solution que nous proposons s'appuie sur la technique d'observation.

L'observation dans ce contexte se doit d'être précise afin de garantir des résultats rigoureux et probants, mais se doit également d'être concise, ciblée, fournissant des informations sans superflus et permettant de construire une représentation pertinente, compréhensible et exploitable du SMA étudié. La représentation, que nous appelons aussi modèle, ainsi obtenue sert directement de base pour l'analyse des performances du SMA.

Parmi les techniques d'évaluation existantes (définies dans la section 2.5 du chapitre 1), le besoin d'obtenir des résultats précis nous oriente vers la technique de mesure. En effet, il s'agit de la technique d'évaluation qui permet d'aboutir à des valeurs exactes, et non à des approximations suite à la position d'hypothèses de simplification tel est le cas de la modélisation analytique ou de la simulation. Ce choix est d'autant plus conforté par notre positionnement initial par rapport à la littérature, qui consiste à s'intéresser aux applications multi-agents. En se plaçant à un niveau applicatif il nous est plus aisé de recourir à la technique de mesure que si on était face à un système physique auquel il serait éventuellement difficile d'accéder.

Ainsi, l'approche d'évaluation des SMA que nous proposons peut se récapituler très sommairement au triplet d'actions « Observer – Modéliser – Mesurer ». A l'image de cette démarche de résolution, l'architecture du système d'évaluation que nous proposons s'articule autour de trois modules :

- **Un module d'observation** qui permet d'observer le comportement du système multi-agents évalué, de collecter les informations pertinentes sur les états et les évènements significatifs et de générer les traces d'exécution en vue de leur analyse ultérieure.

- **Un module de modélisation** qui permet de générer automatiquement un modèle représentatif du SMA étudié en exploitant les traces d'exécution produites par le module d'observation.

- **Un module de mesure** qui permet d'analyser le modèle obtenu et de calculer les mesures associées aux différents critères d'évaluation choisis. Les mesures ainsi obtenues sont ensuite interprétées.

La figure 2.1 illustre l'architecture fonctionnelle globale du système d'évaluation proposé.

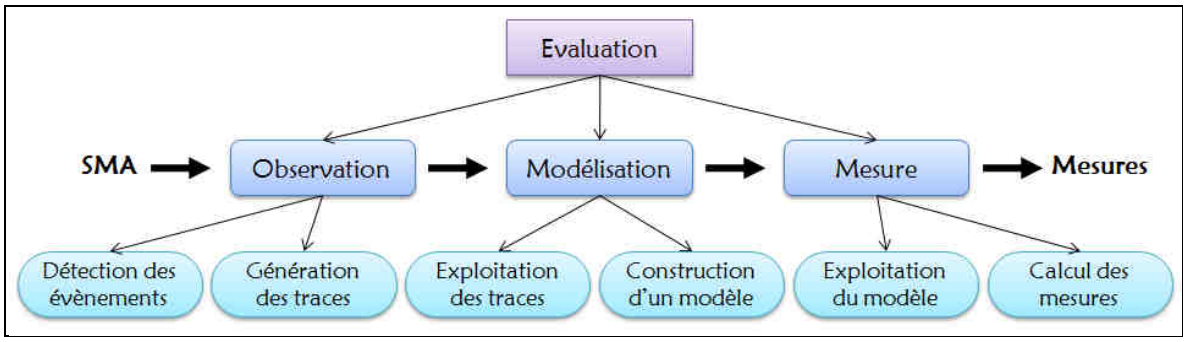

Figure 2.1 : Architecture fonctionnelle du système d'évaluation

Dans la suite de ce chapitre, nous nous intéressons uniquement au module d'observation, les modules de modélisation et de mesure étant détaillés dans le chapitre suivant. Pour ce faire nous commençons par introduire les concepts de base relatifs à l'observation des systèmes.

2. Vue d'ensemble sur l'observation des systèmes

A travers cette section, nous présentons une vue d'ensemble sur l'observation des systèmes informatiques. Il est nécessaire à ce stade de distinguer l'observation dans les systèmes informatiques classiques de l'observation dans les SMA. Bien que la première ait fait l'objet de beaucoup d'études, la deuxième se limite à quelques travaux réalisés dans des perspectives autres que l'évaluation des performances.

2.1. Mise en situation d'observation

D'après Cardon [Car05], une situation d'observation est la situation où un observateur appréhende un certain objet ou phénomène organisé du monde, qui se caractérise par un comportement compliqué. Un observateur est donc un individu (personne ou processus) qui se met dans une situation d'étude minutieuse d'un objet en utilisant des moyens d'analyse dans le but de connaître cet objet à ses différentes échelles d'existence. Pour appréhender le fonctionnement d'un objet organisé, il faut prendre en considération son état interne ainsi que son comportement externe, sachant que les deux se lient et se complètent.

L'observation est une activité incontournable lorsqu'on souhaite obtenir des informations sur l'état ou l'exécution d'un système. Il s'agit d'une étape clé du "monitoring" [Jai91]. Ce dernier est défini comme étant le processus de collecte, d'interprétation et de présentation d'informations se rapportant à un système donné. Toutefois, l'observation se restreint uniquement à la collecte d'informations sur le système et ne procède pas à leur interprétation et présentation. Ces informations concernent les états du système et les évènements qui y ont lieu.

2.2. Concepts de base

A ce stade, il est primordial de définir les concepts de base relatifs à l'observation des systèmes et de présenter la terminologie qui s'y rattache.

➢ **Notions d'état, d'évènement et de trace**

A chaque composant du système observé sont associés des états et des évènements. Du point de vue de l'observation, le comportement du composant est caractérisé par la succession de ses états observés.

- L'état d'un composant est une mesure de son comportement à un instant donné et est représenté par un ensemble de variables d'état contenues dans un vecteur d'état [Ott99].
- Un évènement est défini comme une entité atomique qui représente le reflet d'un changement dans l'état du composant.
- Une trace correspond à la journalisation d'un évènement. Elle comporte souvent l'instant du déroulement de l'évènement, son type et d'autres informations qui lui sont relatives [Jai91].

➢ **Phases de l'observation**

Le processus d'observation d'un système se déroule en deux phases : la phase de collecte des rapports d'états et des rapports d'évènements et la phase de génération de traces.

– *Rapports d'états et Rapports d'évènements* : Le rapport d'état contient un ensemble d'informations sur l'état d'un composant ou d'un groupe de composants du système à un instant donné. Les rapports d'états peuvent être générés périodiquement, à la base d'un calendrier prédéterminé. Ou bien à la demande, c'est-à-dire à la réception d'une requête. Le rapport d'évènements correspond à la détection d'un changement significatif dans l'état d'un objet. On dit qu'un évènement a lieu lorsque certaines conditions prédéfinies sont satisfaites. La détection de l'évènement peut s'effectuer soit à l'intérieur de l'objet par une de ses fonctions, soit à l'extérieur de l'objet par un agent externe qui reçoit les rapports d'états et identifie les changements dans ces états.

– *Génération des Traces* : Afin de décrire le comportement d'un objet durant une période de temps, les rapports d'états et d'évènements sont enregistrés dans l'ordre de leurs occurrences pour constituer les traces d'observation.

➢ **Moyens de l'observation**

Dans ce qui suit nous introduisons la notion de sonde qui représente un moyen à travers lequel l'observation est réalisée. Ce dernier terme étant souvent utilisé pour désigner une entité qui sert à observer un système ou une partie de ce système.

2.3. Types des sondes d'observation

Les sondes d'observation peuvent être classifiées selon le niveau d'implémentation, selon le mécanisme de déclenchement ou bien selon la technique d'observation.

2.3.1. Classification selon le niveau d'implémentation

On distingue trois niveaux d'implémentation possibles pour un observateur [Jai91].

➢ **Sondes matérielles**

Une sonde matérielle consiste en un équipement ou pièce matérielle spécifique dédiée à l'observation et qui se connecte aux composants du système à observer tels que le processeur, les ports, les entrées et les sorties, etc. Généralement, les concepteurs de ce type d'observateurs fournissent pour chaque système une bibliothèque contenant une liste de points où les sondes peuvent s'attacher.

Avantages : Dans le cadre de l'utilisation des sondes matérielles, l'action d'observation est non intrusive. En effet, le système d'observation utilise des ressources autres que celles utilisées par le système observé. Ainsi, l'observation a peu d'effet sur le comportement de ce dernier.

Inconvénients : Cette technique requiert l'installation de matériel additionnel, ce qui est une solution coûteuse. De plus, les informations fournies concernent le niveau le plus bas du système, ce qui n'est pas pratique si l'on désire avoir des informations concernant le niveau applicatif. C'est la classe de mécanismes d'observation la moins portable et la plus compliquée à mettre en œuvre.

- **Sondes logicielles**

Les sondes logicielles sont utilisées pour observer l'activité du système d'exploitation et des logiciels de haut niveau tels que les réseaux et les bases de données. L'application à observer est instrumentée à travers l'insertion de sondes logicielles dans son code. Ces sondes permettent de détecter les évènements significatifs dans l'exécution de l'application.

Avantages : L'observation est orientée application. Les données collectées sont ciblées et faciles à interpréter. De plus, il s'agit d'une solution portable et facilement reproductible sur une large classe d'applications. Les sondes logicielles sont relativement faciles à mettre en œuvre.

Inconvénients : Les sondes logicielles utilisent les mêmes ressources que l'application à observer et donc elles interfèrent avec cette dernière en terme de consommation de temps d'exécution et de mémoire. Ceci peut modifier le comportement du système. Cet impact est d'autant plus perceptible si les informations sont collectées et manipulées en cours d'exécution. Pour limiter cet effet, il faut que l'observation cible les évènements les plus importants.

Il existe plusieurs façons d'utiliser les sondes logicielles :

- Les sondes sont insérées dans le code source du programme à des endroits spécifiques. Cette technique offre la possibilité de placer les points d'instrumentation n'importe où dans le code de l'application. Elle permet aussi d'accéder facilement à l'état des variables internes. L'insertion se fait manuellement, ceci nécessite de disposer du code source de l'application. La procédure est délicate et nécessite la compréhension du fonctionnement du système étudié.

- Les sondes sont insérées dans le code source des bibliothèques de primitives. Les évènements sont détectés par les sondes chaque fois que la bibliothèque correspondante est utilisée. Il s'agit d'une solution plus simple à mettre en œuvre, cependant, uniquement les évènements appartenant à la bibliothèque de primitives peuvent être reportés.
- Les sondes sont insérées au niveau du noyau du système d'exploitation. Elles permettent d'observer les appels système. Les évènements liés à l'application ne sont pas détectés.

➢ **Sondes hybrides**

Les sondes hybrides sont conçues pour bénéficier des avantages des deux autres types de sondes et pour combler leurs manques. Elles possèdent leurs propres ressources et en même temps partagent quelques ressources avec le système observé. Il s'agit typiquement d'un ensemble de sondes logicielles insérées au niveau de l'application avec un matériel dédié pour la collecte et la manipulation des informations observées. De telles sondes sont moins portables mais elles permettent de minimiser l'action intrusive sur le système.

2.3.2. Classification selon le mécanisme de déclenchement

Le processus d'observation peut se déclencher de différentes manières [Jai91], [Man93].

➢ **Périodique**

La technique d'échantillonnage (Time Driving Monitoring) correspond à une observation périodique du système. Les états du système sont observés indépendamment de leurs changements, ce qui implique qu'un même état peut être observé un nombre indéfini de fois. La qualité de la reconstruction de la dynamique de l'exécution est fortement liée au choix du pas d'échantillonnage.

➢ **Événementielle**

L'observation est réalisée lorsqu'un événement d'intérêt est capté par l'environnement d'observation. A chaque occurrence d'un évènement, un enregistrement est réalisé. Un des problèmes majeurs de cette approche est l'impossibilité de prévoir la quantité d'événements à observer.

➢ **A la demande**

L'observation est déclenchée à la demande de l'environnement d'observation. Cette demande peut être périodique ou générée lors de la satisfaction de certaines contraintes.

2.3.3. Classification selon la technique d'observation

Il existe trois techniques possibles communément utilisées pour l'observation.

- **Observation implicite**

Il s'agit de la technique la moins intrusive. Elle consiste à épier le système observé sans avoir à interférer avec lui, ainsi il n'y a pas d'impacts sur ses performance. Cependant, cette technique permet de tout observer, même des informations probablement inutiles. Il faut donc prévoir d'utiliser des filtres spécifiques pour ne garder que les informations qui s'avèrent nécessaires.

- **Observation explicite**

Cette technique consiste à incorporer dans le système observé des points de traces, des capteurs, des compteurs, etc. L'observation explicite cause un certain surcoût vu qu'elle interfère avec le système observé. Cependant, elle permet d'avoir des informations plus ciblées que celles obtenues par observation implicite.

- **Observation par envois de requêtes**

Cette technique consiste à envoyer régulièrement des requêtes au système observé pour l'interroger sur son état actuel. Là encore, une interférence avec le système s'impose mais elle s'avère encore plus importante que la précédente vu que le système lui-même participe activement au processus d'observation, ce qui est susceptible de dérouter son exécution normale.

3. Observation des systèmes multi-agents

Peu de travaux en IA, et notamment en SMA, se sont intéressés à l'observation. Les travaux existants dans ce cadre abordent l'observation dans des perspectives autres que l'évaluation des performances. Nous présentons dans ce qui suit quelques travaux que nous avons recensés dans la littérature.

3.1. Observation d'agents pour des SMA tolérants aux pannes

Les systèmes répartis à large échelle présentent généralement des limites quant à leur résistance aux pannes. La réplication est une solution largement adoptée dans de tels systèmes. Les mécanismes de réplication existants sont statiques étant donné que la criticité des composants peut être déterminée a priori et ne change pas durant l'exécution de l'application distribuée.

Cependant, dans le cas des applications multi-agents, de tels mécanismes s'avèrent insuffisants étant donné que la criticité des agents peut changer durant l'exécution en fonction de l'évolution de leurs comportements.

Dans le cadre du développement d'une plate-forme pour les systèmes tolérants aux fautes, Guessoum et al. [Gue04] proposent une solution pour fiabiliser les agents. Cette solution consiste à leur appliquer automatiquement et dynamiquement les mécanismes de réplication en se basant sur un processus d'observation et un processus de monitoring. L'évaluation de la criticité des agents est basée sur l'observation de leurs exécutions. Le SMA est doté d'un mécanisme d'auto-observation qui permet de déterminer la criticité des agents et ainsi d'adapter dynamiquement la réplication. Les informations observées sont de deux types :

- Informations du niveau système : elles sont basées sur des mesures standard permettant d'évaluer le degré d'activité d'un agent (charge de communication, temps de traitement) ;
- Informations du niveau sémantique : elles dépendent du domaine d'application et du paradigme choisi.

Le mécanisme d'auto-observation proposé est distribué. Cette distribution est fondée sur une organisation d'agents réactifs appelés Moniteurs dont le but consiste à observer et contrôler les agents du domaine. A chaque agent du domaine est associé un agent Moniteur et à chaque site est associé un agent Observateur tel que le montre la figure 2.2.

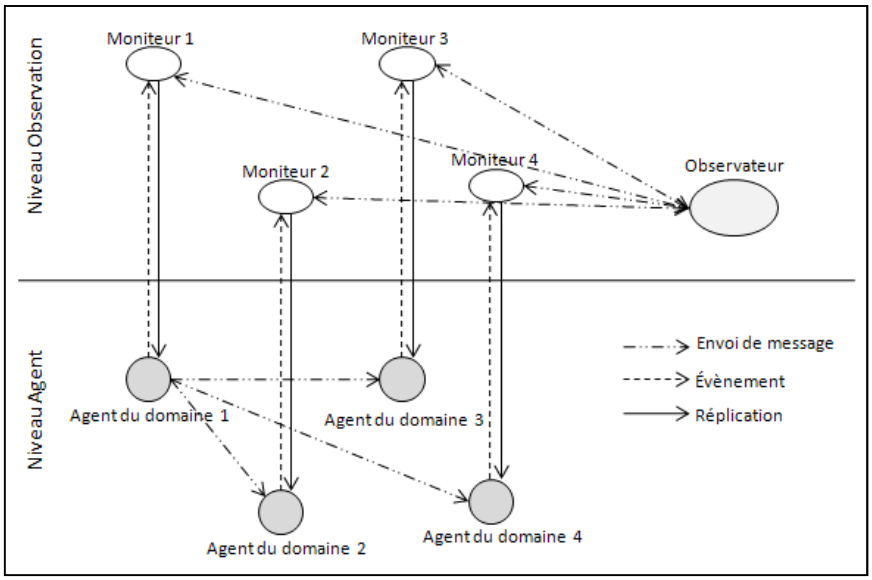

Figure 2.2 : Architecture d'observation multi-agents [Gue04]

L'observation des agents se fait grâce à un module d'observation intégré aux serveurs de la plate-forme présents sur chaque machine hôte. Les informations fournies par ce module sont gérées par l'agent Observateur. Chaque agent Moniteur s'abonne à ce dernier pour recevoir les informations et les événements correspondant à l'agent du domaine auquel il est lié. Il analyse ensuite ces informations pour évaluer la criticité de l'agent du domaine et mettre à jour sa réplication.

Les agents de contrôle (Moniteurs et Observateurs) sont organisés de façon hiérarchique. Chaque agent Moniteur ne communique qu'avec l'agent Observateur. Les agents Observateurs s'échangent des informations locales afin d'en déduire des informations globales du système (nombre de réplicas global, quantité d'informations échangée entre agents, charge de communication moyenne, nombre de messages moyen, etc.).

3.2. Observation répartie des interactions dans les SMA

L'interaction entre les agents est une caractéristique très importante des SMA. Les interactions se définissent généralement comme étant des composantes principales et essentielles de la dynamique du SMA. Elles permettent aux agents de coopérer, de négocier ou de participer à la satisfaction d'un but global. En général, les interactions sont mises en œuvre à l'aide d'un mécanisme de transfert d'informations entre les agents participants, à savoir la communication.

El Fallah et al. [Elf98] s'intéressent à l'étude et l'analyse des résultats des interactions au sein d'un SMA afin d'améliorer son comportement lors des interactions futures. Pour ce faire, ils proposent une approche générique, indépendante du langage d'interaction utilisé, combinant deux paradigmes : l'observation répartie, pour capturer les événements liés aux interactions, et la modélisation par les réseaux de Pétri en vue de leur analyse. La figure 2.3 illustre l'architecture fonctionnelle de toute la démarche d'analyse.

Le processus d'observation et d'analyse des interactions se déroule comme suit :

- Chaque agent dispose d'un module d'observation local qui garde une trace à chaque fois que l'agent exécute un événement observable (ex. émission ou réception de message).
- Tous les autres agents, participant à l'observation répartie, envoient la trace de leur exécution à un agent observateur central localisé sur un des sites. Ce dernier se charge de l'analyse des traces, permettant ainsi de construire une vue globale du comportement du système.

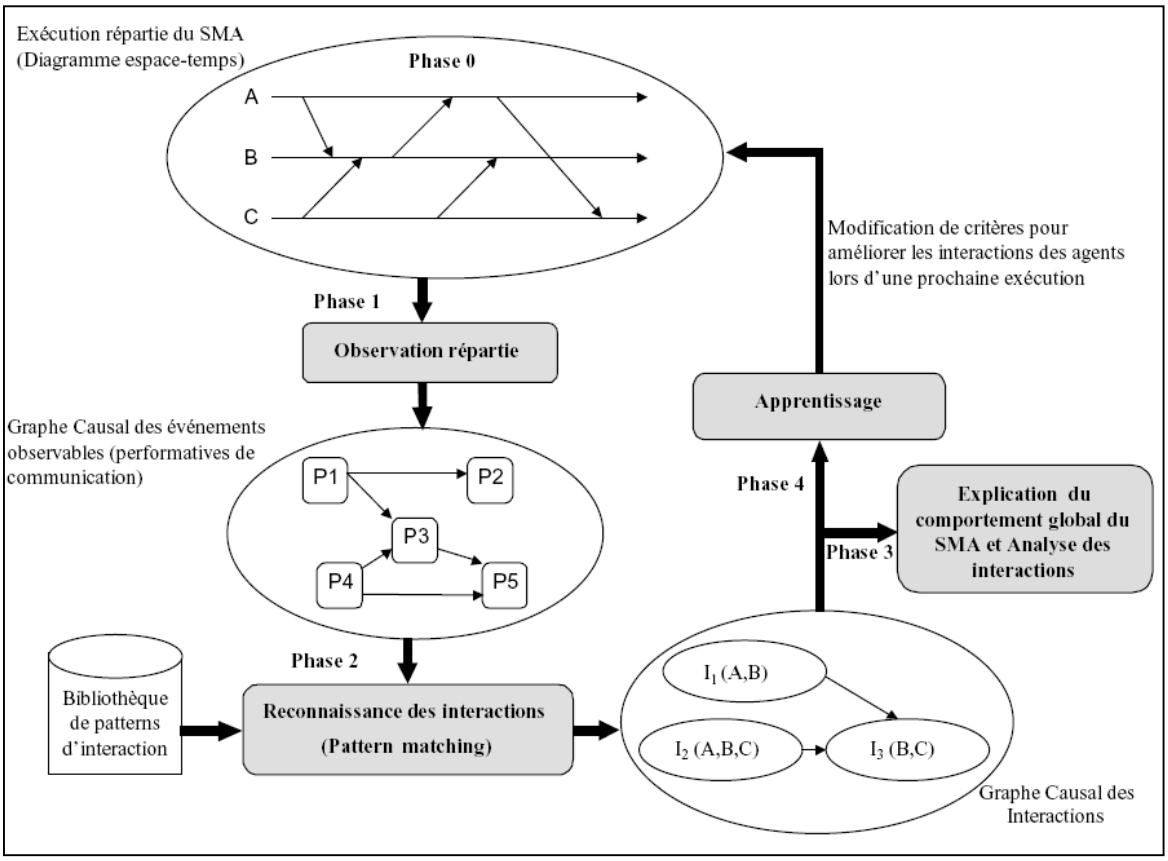

Figure 2.3 : Architecture fonctionnelle de la démarche [Elf98]

3.3. Observation et contrôle morphologique dans les SMA massifs

Dans le cadre de ses travaux sur la conscience artificielle, et plus précisément dans le cadre de la modélisation et de la conception d'un système générateur d'émotions et de pensées artificielles, Cardon [Car04] avance l'hypothèse que ces dernières peuvent être modélisées en utilisant des SMA massifs auto-adaptatifs.

Partant du constat que les méthodes de contrôle usuelles basées sur le monitoring de chaque agent séparément ne conviennent pas à un SMA massif de l'ordre de 10^2 à 10^5 agents. Campagne et al. [Cam03] proposent un modèle basé sur la notion de morphologie qui est un moyen dynamique pour contrôler l'organisation d'un système d'agents massif. Le système de contrôle morphologique proposé ne s'intéresse pas aux états internes des agents mais plutôt à leurs organisations. Le modèle multi-agents proposé qui en fait usage est décrit par la figure 2.4.

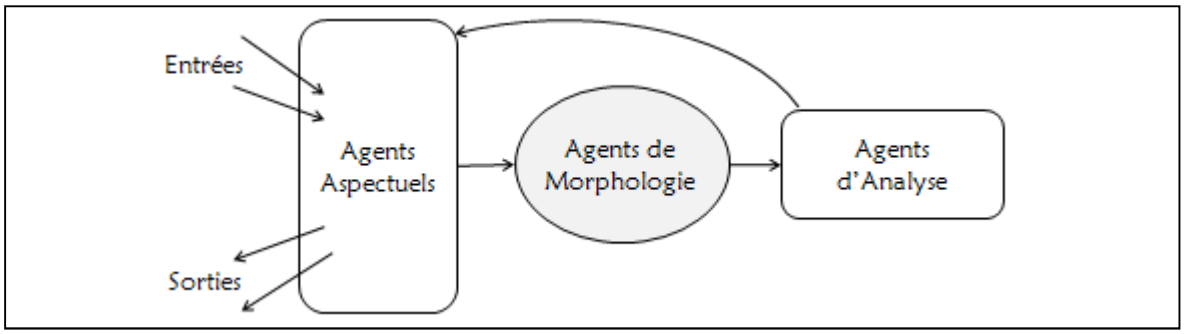

Figure 2.4 : Modèle multi-agents basé sur le contrôle morphologique [Car04]

Le modèle proposé se compose de trois organisations d'agents :

- *L'organisation aspectuelle* qui se compose des agents aspectuels. Il s'agit du SMA destiné soit à représenter un phénomène, soit à piloter un système tel un robot. Il convient de mentionner ici qu'un agent aspectuel est un agent logiciel qui exprime un caractère ou un rôle fonctionnel précis [Car04].
- *L'organisation morphologique* qui se compose d'agents morphologiques. Elle permet de décrire la structure de l'organisation aspectuelle et sa dynamique. Ceci se fait en observant constamment l'évolution des grandeurs qui caractérisent les agents aspectuels telles que les flots de données qu'ils manipulent et produisent, les processus qu'ils activent, les communications qu'ils effectuent, etc. et de les représenter par des formes qui seront analysées ultérieurement. Campagne présente dans sa thèse [Cam05] plusieurs propositions de formes qui peuvent caractériser une organisation d'agents.
- *L'organisation d'analyse* qui se compose d'agents d'analyse. Cette organisation permet d'assurer le contrôle effectif, elle prend en compte la description morphologique pour faire le pilotage de l'organisation aspectuelle.

Jusque là, nous avons présenté les notions de base relatives à l'observation. La question qui se pose à présent est : "Quelle approche doit-on utiliser pour aborder le problème de l'observation des SMA en vue d'évaluer leurs performances ?".

A travers cette partie, nous essayons donc de répondre à cette question en formalisant notre approche de résolution. Nous commençons par étaler les différents choix conceptuels qui se présentent, puis nous décrivons les détails de la solution proposée au vu des choix effectués.

4. Choix conceptuels pour la solution d'observation

Dans cette section, nous exposons de manière détaillée tous les choix qui se présentent en vue de spécifier et de concevoir notre solution pour l'observation des SMA.

4.1. Objectifs et besoins de l'observation

L'observation se voit souvent répondre à un ou plusieurs objectifs dictés par l'intérêt de l'observateur. En ce qui nous concerne, nous devons toujours garder en perspective que l'idée de construire un système d'observation est avant tout motivée par une optique d'évaluation de performances. Pour cette raison, la solution proposée doit répondre à un certain nombre de besoins dont principalement :

- **La performance** : la solution doit être assez légère, c'est-à-dire qu'elle doit minimiser les impacts sur la performance du système observé. D'après Jain [Jai91] une surcharge de moins de 5% est considérée comme très adéquate, sachant qu'une surcharge de plus de 100% est possible.
- **L'efficacité** : la solution doit assurer le filtrage des informations observées de manière à ne sauvegarder que le strict nécessaire pour l'évaluation. Ceci permettra d'abord de diminuer la surcharge induite par l'observation et en plus d'optimiser les capacités de stockage des informations de traces.
- **La réutilisation** : les observateurs utilisés doivent être réutilisables étant donné que nous visons la construction d'un système d'évaluation générique et utilisable pour la plus large classe possible d'applications multi-agents.
- **La flexibilité** : la solution doit se comporter de manière souple face aux besoins d'évolution éventuels. Elle doit, par exemple, supporter la prise en compte de nouvelles informations à observer. Elle doit aussi permettre d'activer ou de désactiver le mécanisme d'observation selon le besoin.

4.2. Modélisation du système observé

Avant de procéder à l'observation d'un système, il est important de commencer par le modéliser. Cette modélisation permettra de mieux expliciter sa structure et de déterminer les aspects les plus importants dont on doit tenir compte lors de l'observation. Nous avons identifié deux approches de modélisation possibles qui peuvent être combinées : une approche par découpage en composants et une approche par découpage en niveaux d'abstraction.

4.2.1. Découpage en composants

Un composant peut être soit un élément primaire tel qu'une ressource, un objet ou un agent, soit un élément composé lui même de composants primaires. Un composant est une structure dynamique qui encapsule un état et un comportement. Le concept de composant permet donc de modéliser les éléments manipulés dans un système distribué et notamment dans un SMA. Ainsi, l'approche de découpage en composants consiste à décomposer le système d'une manière "horizontale". Dans ce cas, chaque composant est observé à part, et l'état de tout le système est déduit à partir des observations effectuées sur tous ses composants.

4.2.2. Découpage en niveaux

Tout système informatique est communément modélisé par la superposition verticale de trois niveaux : applicatif, système et matériel.

- Le niveau applicatif représente les applications de l'utilisateur. Il réalise des opérations de haut niveau en utilisant des abstractions des niveaux inférieurs.
- Le niveau système présente les services nécessaires à l'exécution des applications de l'utilisateur et définit les méthodes d'accès et d'utilisation du niveau matériel.
- Le niveau matériel représente les ressources physiques du système mises à la disposition du niveau supérieur.

L'approche multi-niveaux repose donc sur le découpage du système observé en niveaux d'abstraction d'une manière "verticale". Ce découpage a pour objectif de faciliter l'interprétation des phénomènes observés.

4.2.3. Choix de la modélisation

Dans le but d'assurer une modélisation adéquate à notre problème d'observation, nous proposons de combiner les deux types de découpage. En effet, nous optons pour une modélisation multi-niveaux de notre SMA. Ce dernier peut être vu, comme tout autre système informatique, comme une superposition de trois couches :

- La couche la plus basse définit le matériel sur lequel s'exécute le SMA (un ordinateur, un réseau informatique, etc.)
- La couche intermédiaire définit à la fois le système d'exploitation ainsi que la plate-forme d'exécution du SMA. Cette dernière représente les différents services et

mécanismes mis à disposition des agents pour être opérationnels (service de gestion des agents, mécanismes de communication, etc.)

– La couche la plus haute définit l'application multi-agents de l'utilisateur avec toutes ses entités telles que spécifiées le concepteur de l'application.

Nous proposons, suite à cela, de faire abstraction des niveaux bas et intermédiaire pour ne plus considérer que le niveau applicatif. C'est-à-dire que nous nous dégageons des contraintes matérielles et des implémentations physiques du SMA lors de son évaluation et par conséquent de son observation.

Par ailleurs, et pour mieux expliciter la structure du SMA, nous procédons aussi à un découpage en composants de la couche applicative. En effet, nous considérons notre SMA comme une agrégation d'entités de base qui sont les agents. De ce fait, nous émettons l'hypothèse que l'observation d'un SMA doit passer nécessairement par l'observation des différentes entités qui le composent à leur niveau d'abstraction le plus élevé.

4.3. Architecture d'observation

L'architecture du module d'observation adoptée aura un impact immédiat sur ses performances. Pour cela, nous sommes amenés à étudier toutes les alternatives possibles pour choisir celle qui répond le mieux à nos attentes. Nous commençons donc par étudier les avantages et les inconvénients d'une solution centralisée puis celles d'une solution distribuée pour la réalisation de notre module d'observation.

4.3.1. Architecture d'observation centralisée

Dans ce type d'architecture (voir Figure 2.5.a), un seul processus/entité se charge d'observer tous les agents, de collecter les informations et de générer les traces. L'observateur interroge périodiquement les agents sur leurs états. Les interactions entre l'observateur et les agents observés s'expriment par des envois et des réceptions de messages de type requête/réponse. Le même processus d'observation se charge aussi de générer les traces qui seront ensuite exploitées par le module de mesure pour calculer les mesures de performances.

4.3.2. Architecture d'observation distribuée

Dans ce type d'architecture (voir Figure 2.5.b), plusieurs entités sont disséminées à travers l'ensemble du SMA. A chaque agent est associé un observateur local qui contrôle son exécution et intervient à chaque détection d'un événement significatif pour générer une

trace. Les traces d'exécution des différents agents sont donc distribuées et doivent être collectées par une autre entité qui se charge de reconstruire la trace d'exécution globale de tout le SMA.

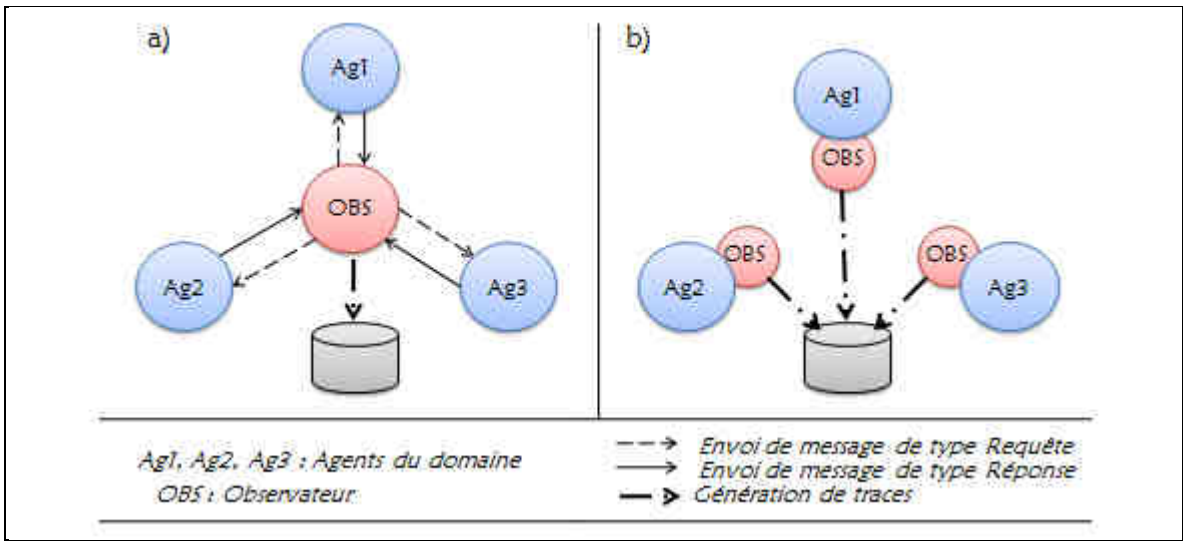

Figure 2.5 : Observation Centralisée vs. Observation Distribuée

4.3.3. Choix de l'architecture

L'avantage d'une observation centralisée c'est qu'elle permet de reconstruire une image globale de l'exécution du système. Cependant, elle provoque une surcharge au niveau du processus d'observation et induit des coûts de communication supplémentaires. De plus, les agents observés sont impliqués de manière explicite dans l'observation ce qui engendre des temps de latence dans leur traitements et par conséquent affecte leurs performances.

L'avantage d'une architecture distribuée c'est qu'elle permet une observation assez fine du SMA en dotant chaque agent d'un module d'observation et de traçage local. De plus, en cas de défaillance d'un des observateurs, il est toujours possible de reconstruire la trace d'exécution globale à partir d'une information incomplète. Cependant, un nombre trop important d'observateurs peut alourdir le système en consommant davantage de ressources.

Vu les avantages qu'offre la distribution, nous proposons d'adopter une architecture d'observation distribuée basée sur des sondes logicielles. Chaque agent est doté d'une sonde logicielle réactive qui observe son exécution et génère une trace chaque fois qu'elle détecte un évènement significatif dans son comportement.

4.4. Utilisation des sondes logicielles

Dans cette section, nous présentons les différents types de sondes logicielles que nous pouvons utiliser.

4.4.1. Sondes orientées caractéristique

A chaque caractéristique que nous souhaitons évaluer, nous concevons une sonde appropriée qui se charge de collecter les informations et les données nécessaires à la mesure de cette dernière. Chaque sonde agit indépendamment des autres et ne détecte que les évènements nécessaires à l'évaluation d'une caractéristique préalablement fixée. Par exemple, pour évaluer les interactions dans le SMA, la sonde ne détectera que les actions relatives aux interactions telles que les échanges d'informations entre agents.

Dans le cas où nous souhaitons évaluer deux caractéristiques dont les mesures nécessitent de collecter les mêmes informations, ces dernières seront collectées deux fois par les deux sondes adéquates et il se trouve que nous aurons la même information dupliquée, ce qui induit d'abord un problème de cohérence (à quelle valeur de la même information se fier ?) et en plus, un coût très élevé de l'observation. Toujours dans le même cas où nous souhaitons évaluer N caractéristiques, nous devrions instancier N sondes multipliées par le nombre d'agents du SMA (une sonde pour chaque agent) ce qui fait que nous aurons plus de sondes que d'agents. Cette solution présente beaucoup d'inconvénients, elle est donc à écarter.

4.4.2. Sondes auto-adaptatives

Une sonde auto-adaptative, comme son nom l'indique, s'adapte en fonction de la caractéristique à évaluer pour ne détecter que les évènements adéquats et ne collecter que les informations pertinentes. Ainsi, nous n'aurons à concevoir qu'un seul type de sondes pour toutes les caractéristiques.

Cependant, dans le cas où nous souhaitons évaluer plus d'une caractéristique, la sonde s'adapte pour procéder à la collecte des informations nécessaires à l'évaluation de la première caractéristique. Ensuite, elle effectue le même procédé pour la caractéristique suivante, et ainsi de suite. L'inconvénient majeur de ce type de sondes, c'est qu'elles nécessitent à chaque fois une phase d'adaptation, ce qui ralentit le processus d'observation. De plus, la collecte des informations relatives à chaque caractéristique s'effectue séquentiellement. Il s'avère donc que ce type de sonde n'est pas performant et donc son utilisation est à écarter.

4.4.3. Sondes paramétrables

Une sonde paramétrable est une sonde qui permet de détecter les évènements et de collecter les informations relatives à plus d'une caractéristique à la fois. En effet, pour réaliser une sonde paramétrable, il suffit de réaliser un seul modèle générique pour cette sonde qui permet de détecter tous les évènements possibles et de collecter toutes les informations sur l'état d'un agent. Selon les caractéristiques à évaluer -que la sonde prend en paramètres- elle active simultanément les parties nécessaires à l'observation de ces caractéristiques.

L'avantage de ce type de sondes, c'est que nous n'avons qu'à concevoir un seul modèle de sonde qui soit paramétrable en fonction des caractéristiques à évaluer. Ainsi, lors de l'exécution du SMA, nous instancions autant de sondes que d'agents sans se soucier du nombre de caractéristiques concernées par l'observation.

4.4.4. Choix du type des sondes

Vu les avantages de généricité et de possibilité d'adaptation selon les caractéristiques à évaluer, nous optons pour l'utilisation des sondes paramétrables. En effet, ces dernières offrent plus de souplesse quant au filtrage des informations à tracer. Une description plus détaillée du fonctionnement de ces sondes est décrit ultérieurement au cours de ce chapitre.

5. Solution proposée

Étant donné les besoins auxquels doit répondre le module d'observation proposé, et au vu des objectifs que nous nous sommes fixés dès le départ, nous estimons que la programmation orientée-aspects offre un cadre propice au développement de notre solution.

5.1. La Programmation Orientée Aspects

Nous consacrons cette partie à la présentation de ce paradigme relativement nouveau, tout en cadrant les notions les plus importantes dont nous aurons besoin.

5.1.1. Principe et notions de base

Le développement d'un logiciel est souvent contraint par la satisfaction de plusieurs objectifs qui ne relèvent pas forcément de ses besoins fonctionnels. On appelle de tels objectifs des *Préoccupations* (en anglais, Concerns). Les concepteurs et les développeurs

visent à séparer au mieux les diverses préoccupations d'une application pour assurer leur éventuelle réutilisation. Il s'agit là d'un problème bien connu en génie logiciel, appelé la *Séparation des Préoccupations* (en anglais, Separation of Concerns) [Dij82].

Les méthodologies de conception et de développement existantes, telles que la méthodologie orientée objet, ont essayé tant bien que mal d'apporter des éléments de réponse à cette question en imposant une décomposition fonctionnelle et modulaire de l'architecture du logiciel à produire. Cependant, elles présentent toujours des limites surtout lorsque ces préoccupations sont étroitement liées et enchevêtrées, et par conséquent difficiles à dissocier et à intégrer dans des modules indépendants.

La Programmation Orientée Aspects : POA (en anglais, Aspect Oriented Programming) permet précisément d'apporter une solution en même temps novatrice et très puissante au problème de séparation des préoccupations lors du développement d'une application. Introduite en 1997 par Gregor Kiczales et al. [Kic97], la POA consiste à dissocier la logique métier et les fonctions transverses ou orthogonales (en anglais, Crosscutting Concerns) telles que la persistance, la sécurité, l'authentification, la synchronisation, etc. Il s'agit donc de scinder l'application de manière à dégager :

- Les aspects métier fonctionnels qui réalisent les services et les fonctionnalités de base de l'application ;
- Les aspects techniques non fonctionnels qui régissent l'exécution de l'application de façon transversale.

L'objectif étant de définir ces derniers dans des entités logicielles découplées, indépendantes, autonomes et réutilisables appelées *Aspects* (voir Figure 2.6).

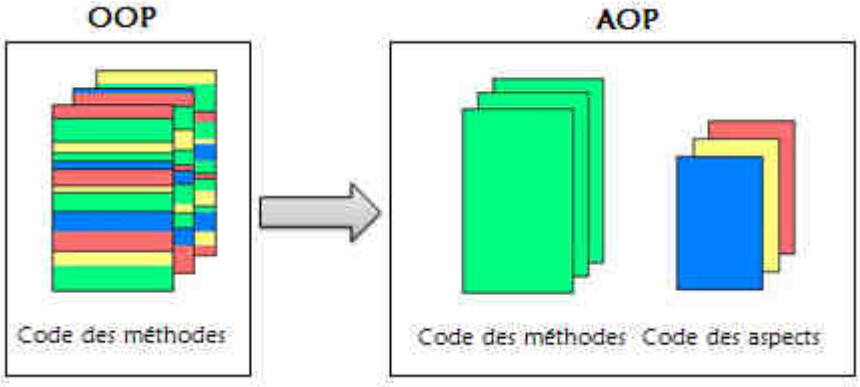

Figure 2.6 : Séparation des préoccupations en Programmation Orientée Aspects [Lad02]

La construction d'une application orientée aspects passe par trois étapes différentes [Lad02], tel que le montre la figure 2.7 :

- Décomposer les besoins de l'application et identifier les différents aspects fonctionnels (logique métier) et les aspects non fonctionnels (préoccupations techniques) ;
- Programmer chacun des aspects identifiés séparément ;
- Recomposer l'application en intégrant les différents aspects ainsi programmés.

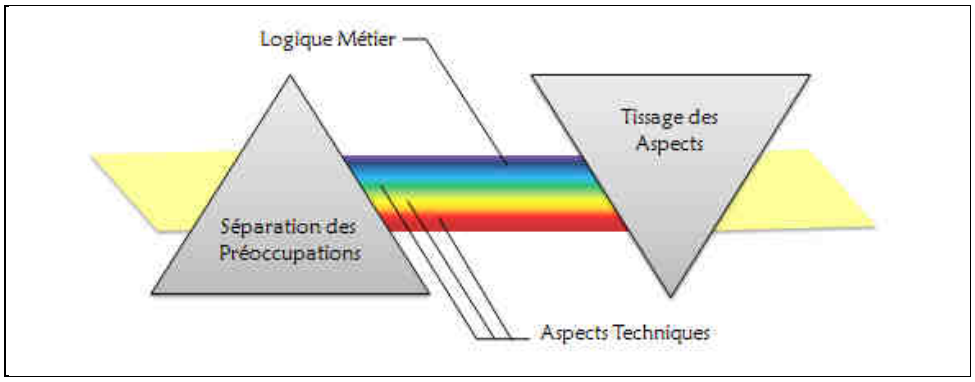

Figure 2.7 : Principe de la Programmation Orientée Aspects [Lad02]

La construction de l'application à partir des aspects nécessite une phase d'assemblage pendant laquelle l'aspect de base et les aspects techniques sont intégrés. En POA, ce mécanisme s'appelle *Tissage* (en anglais, Weaving). Le tissage de l'application consiste à modifier le flot d'exécution de l'aspect de base de manière à prendre en compte les aspects techniques. L'interception du flot d'exécution de l'aspect de base s'effectue à des endroits bien définis de ce dernier appelés *Points de Jonction* (en anglais, Join Points). Lorsqu'un point de jonction est atteint, le programme correspondant à un aspect technique déterminé est activé, on appelle ce programme un *Greffon* (en anglais, Advice). Ainsi, un Aspect au sens de la POA est un module défini par la donnée de greffons et de points d'activation de ces greffons, appelés encore *Points de Coupure* (en anglais, Pointcut).

Il est à noter qu'il existe deux grandes stratégies de tissage d'aspects [Bou01] :

- Le tissage statique consiste à instrumenter l'application en référençant les greffons lors de la phase de compilation, ce qui fait que le surcoût engendré en temps d'exécution sur le programme est assez faible.
- Le tissage dynamique consiste à tisser les aspects lors de la phase d'exécution. Cette stratégie de tissage est malheureusement très coûteuse en temps d'exécution du programme.

5.1.2. La POA et la réflexion

La POA s'apparente beaucoup, dans son principe, à la notion de réflexion qui permet elle aussi de séparer les préoccupations d'une application en la décomposant en deux niveaux : un niveau de base et un niveau méta. Le niveau de base se compose d'objets qui réalisent la logique métier de l'application, alors que le niveau méta se compose de méta-objets qui interprètent et manipulent les entités fonctionnelles du niveau de base en les réifiant (voir Figure 2.8). En réflexion, un seul langage réflexif est utilisé pour décrire le niveau de base et le niveau méta, alors qu'en POA les aspects techniques sont décrits par un langage dédié, généralement une extension du langage dans lequel est écrite la logique applicative. Un système doté d'une architecture réflexive a la capacité de représenter, d'observer et de modifier sa structure et son propre comportement [Smi84], [Mae87].

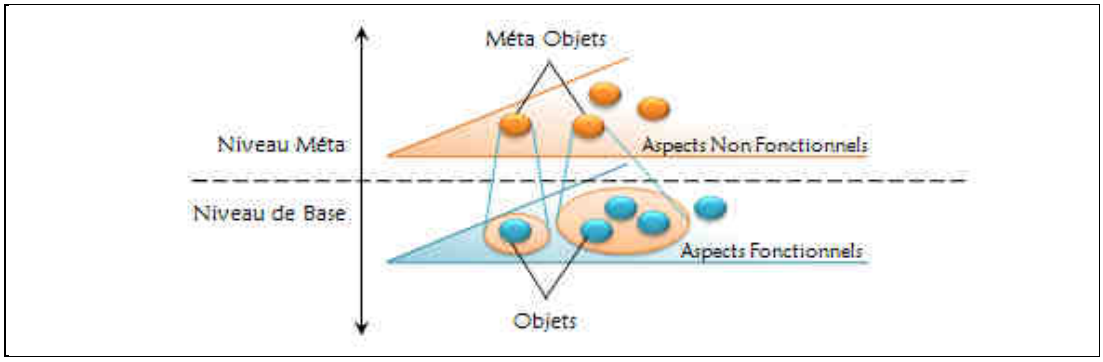

Figure 2.8 : Architecture d'un système réflexif

5.1.3. Apports et utilisations de la POA

La POA a permis de réduire considérablement les problèmes induits par l'embrouillement et la dispersion des propriétés techniques en se basant sur le principe de la séparation des préoccupations. L'avantage de cette technique est qu'elle peut s'utiliser conjointement avec n'importe quelle méthodologie de développement et avec n'importe quel langage de programmation, pourvu qu'une implémentation existe. Ceci n'est pas l'unique avantage que l'on peut en tirer. En effet, grâce à la POA, le couplage entre les modules gérant les aspects techniques peut être réduit de façon très importante, du moment qu'elle permet d'adresser chaque préoccupation de manière indépendante. Ceci permet d'atteindre une meilleure modularité et réutilisation du logiciel. Ainsi, chaque module, représenté par un aspect, peut être réutilisé indépendamment de l'environnement ou du domaine d'application. Par ailleurs, la POA permet de produire des logiciels dont la maintenance est aisée, elle permet aussi au développeur de gagner en productivité en augmentant le parallélisme du développement.

Les utilisations possibles de la POA sont nombreuses. Smacchia et al. [Sma03] en présentent un panorama dans leur article dédié aux intérêts et usages de la POA. Une des utilisations les plus citées de la POA est le traçage de l'exécution d'une application. Il s'agit d'une tâche assez coûteuse pour les développeurs étant donné qu'elle nécessite d'intervenir au niveau du code source pour y insérer des points et des primitives de trace. Ces primitives sont pratiquement les mêmes pour tous les modules de l'application à tracer, ce qui induit une réplication d'une bonne partie du code. De plus, il se trouve que ces primitives soient éparpillées dans le code métier de l'application, ceci rend leur maintenance et leur désactivation (si nécessaire) très difficiles. La solution que suggère la POA consiste à définir ces primitives dans des aspects indépendants qui seront activés si besoin est. A la manière du traçage d'exécution, la POA s'utilise pour assurer les fonctions d'authentification et de gestion des utilisateurs. Elle s'utilise aussi pour réaliser les fonctions d'archivage de données, de synchronisation des accès concurrents à une ressource, d'utilisation d'objets distants, etc.

5.1.4. Utilisation de la POA dans le cadre des SMA

Depuis quelques années, quelques travaux de recherche, initiés dans des perspectives d'application de la POA dans la méthodologie orientée agents, ont vu le jour. Ces travaux tentent tous d'améliorer le processus de développement des SMA en tirant profit des avantages qu'offre la POA.

D'après le recensement des travaux que nous avons pu recueillir de la littérature, l'exploration de cet axe de recherche a été entamée par Garcia et al. en 2002 [Gar02], [Gar06]. L'essentiel de ces travaux consiste à utiliser la POA pour isoler les propriétés comportementales des agents telles que l'autonomie, l'interaction, l'adaptation, la collaboration, l'apprentissage et la mobilité. Chacune de ces propriétés est encapsulée dans un aspect à part. L'objectif de cette approche consiste d'une part à faciliter le développement des SMA, et d'autre part, à améliorer la maintenance et la réutilisation de leurs fonctionnalités.

Un peu plus tard, et parallèlement aux travaux de Garcia et al., Robbes et al. [Rob04] cherchent à importer les concepts de la POA dans les modèles des SMA. Leur étude porte sur le méta-modèle Aalaadin [Fer98]. Ce méta-modèle permet de simplifier la conception des SMA en proposant les concepts Agent, Groupe et Rôle (AGR). Les travaux de Robbes et al. [Rob04] consistent à unifier les notions de groupe et d'aspect en s'appuyant sur la réification des rôles. Cette extension d'Aalaadin a pour objectif de permettre la

réutilisation des rôles individuellement en les découplant des définitions des agents. Ainsi, la modularité des groupes, définis initialement comme des ensembles de rôles, est augmentée considérablement.

Par ailleurs, et dans la même perspective des travaux cités ci-dessus, Amiguet et al. proposent une approche orientée aspects pour la conception d'applications orientées agents appelée MOCA [Ami04]. MOCA est un modèle multi-agents basé sur le méta-modèle Aalaadin. Sa structure se compose de deux niveaux conceptuels : le niveau organisationnel, qui décrit l'organisation des agents en termes de groupes et d'interactions entre ces groupes, et le niveau multi-agents qui spécifie les agents comme entités primaires de décomposition. Chacun de ces deux niveaux, agents et organisations, est vu comme un aspect indépendant et est conçu et développé séparément, garantissant ainsi une meilleure réutilisation.

5.2. Description de la solution proposée

Maintenant que nous avons introduit la programmation orientée aspects, nous dédions cette section à la présentation de notre solution pour l'observation des SMA en tirant profit des atouts que nous offre la POA.

5.2.1. Architecture globale du module d'observation

En se basant sur la notion de séparation des préoccupations introduite par la POA, et par analogie au concept de réflexivité, nous considérons que l'observation est une préoccupation technique orthogonale à la logique applicative du système à observer. Nous proposons donc de définir cette fonctionnalité dans une couche à part, appelée *couche d'observation*. Cette couche se superpose à la couche qui réalise les fonctions de base du SMA, appelée *couche fonctionnelle* (voir Figure 2.9).

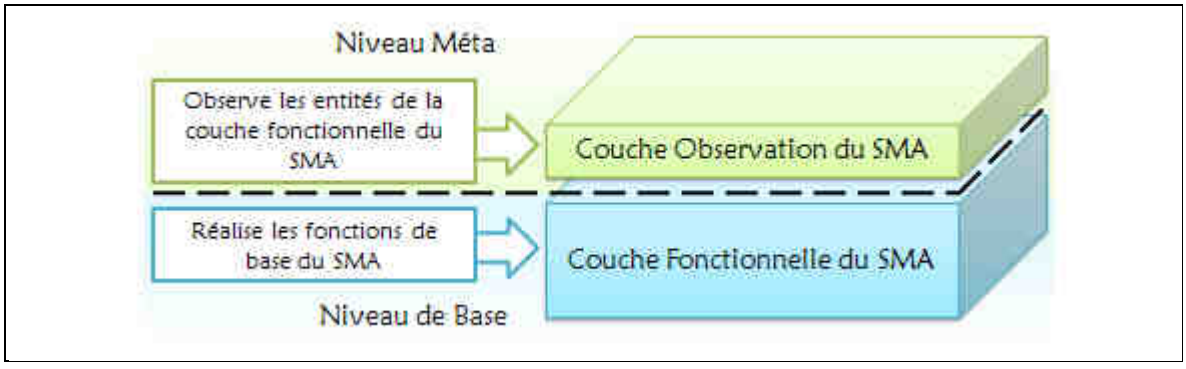

Figure 2.9 : Séparation des couches fonctionnelle et d'observation du SMA

La figure 2.10 illustre de manière plus détaillée les composantes de chacune de ces deux couches.

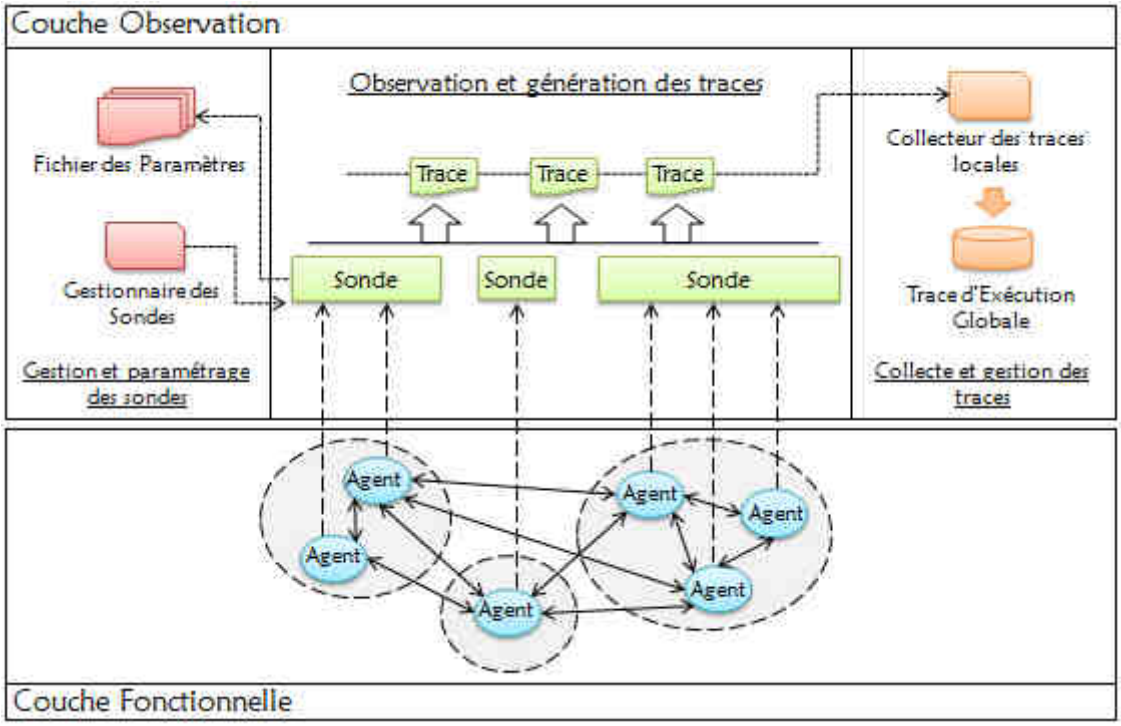

Figure 2.10 : Architecture détaillée du module d'observation

La couche fonctionnelle se compose des agents du SMA qui peuvent s'exécuter sur des sites physiquement distribués. La couche d'observation, quant à elle, se compose de différentes entités qui coopèrent en vue d'assurer la tâche d'observation. Ces entités peuvent être regroupées en trois niveaux selon les services qu'elles offrent :

- La gestion et le paramétrage des sondes ;
- L'observation et la génération des traces ;
- La collecte et la gestion des traces.

Dans ce qui suit, nous détaillons les fonctionnalités de chacun de ces services.

5.2.2. Le service de gestion et de paramétrage des sondes

Il est assuré par deux entités :

- Le gestionnaire des sondes : il s'agit d'un agent qui permet de gérer les sondes d'observation de manière transparente à l'utilisateur, il assure principalement leur activation ou désactivation selon le besoin. Il permet aussi de paramétrer les sondes

en vue de l'observation des informations relatives aux caractéristiques souhaitées. Le paramétrage s'effectue en se basant sur un fichier de paramètres.

– Le fichier des paramètres : il contient l'association entre les caractéristiques des SMA et les informations nécessaires à leur évaluation.

5.2.3. Le service d'observation et de génération des traces

Il est assuré par les sondes d'observation. Il s'agit sondes logicielles paramétrables définies dans aspects, dont le fonctionnement consiste en l'enchaînement des étapes suivantes :

– Le paramétrage : il s'effectue une seule fois au début de l'exécution de la sonde. Cette dernière active les parties nécessaires à l'observation des informations relatives aux caractéristiques que l'on souhaite évaluer. Par exemple, il serait inutile d'observer et de journaliser les envois et réceptions de messages si nous allons évaluer une caractéristique qui n'en fait pas usage. Une fois la sonde est active, elle s'engage dans un comportement cyclique du type détection/action.

– La détection : elle est assurée par le mécanisme de contrôle du flot d'exécution qu'exerce le tisseur d'aspects. En effet, ce dernier permet de déclencher l'action adéquate chaque fois qu'un évènement d'intérêt est capté, ce qui se traduit par l'atteinte d'un point de jonction spécifique. Un évènement peut être soit :
  - l'exécution d'une action sur un objet ou une ressource de l'environnement ;
  - l'exécution d'une action sur lui-même (la modification de son état interne) ;
  - l'interaction avec un autre agent ;
  - l'envoi ou la réception d'un message.

– L'action : lorsqu'un point de jonction est atteint, le greffon correspondant est lancé en exécution. Un greffon définit donc l'action à effectuer lors de la détection d'un évènement. Il s'agit, dans notre cas, de la génération d'une trace. En d'autres termes, le greffon est une portion de code qui permet de sauvegarder les rapports d'états et d'évènements de l'agent observé. Les traces d'exécution doivent se conformer à une structure prédéfinie pour faciliter leur manipulation et éventuellement leur utilisation par le module de mesure. Nous proposons donc que les traces générées soient au format XML.

La figure 2.11 illustre le principe de fonctionnement de la sonde paramétrable.

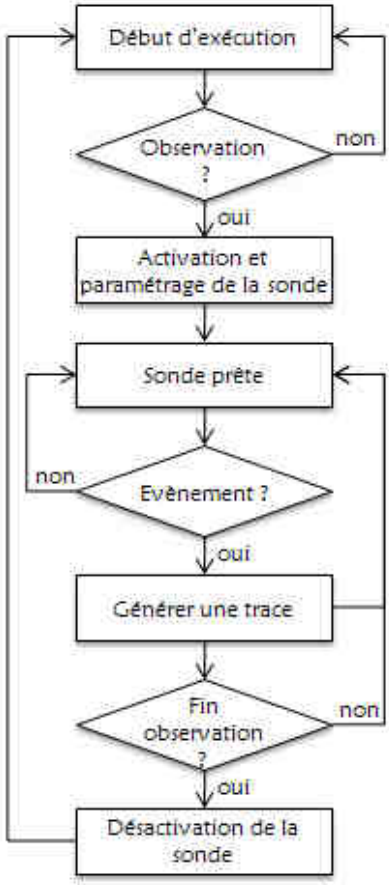

Figure 2.11 : Organigramme représentant l'algorithme de la sonde paramétrable

5.2.4. Le service de collecte et gestion des traces

Il est assuré par un agent qui collecte les traces locales et les fusionne pour créer une trace globale de l'exécution du SMA. Cette trace permettra d'avoir une vue globale de ce qui s'est passé dans tout le SMA malgré la distribution physique et facilitera considérablement le processus de mesure.

Conclusion

Dans ce chapitre, nous avons proposé une approche d'évaluation des SMA basée sur l'observation. L'architecture du système d'évaluation proposé se compose de trois modules : un module d'observation, un module de modélisation et un module de mesure. Nous avons ensuite détaillé la solution proposée pour l'observation des SMA qui consiste à définir des sondes logicielles paramétrables basées sur la programmation orientée-aspects. L'architecture détaillée du module d'observation ainsi que le mode de fonctionnement des sondes ont été exposés à cet effet.

CHAPITRE 3

UNE APPROCHE DE MESURE DES PERFORMANCES DES SMA BASÉE SUR LA MODÉLISATION

Sommaire

A travers ce chapitre, nous définissons une approche de mesure des performances des SMA basée sur la modélisation. Cette approche s'intègre dans le cadre de la solution globale proposée pour l'évaluation des SMA dont la première étape -l'observation- a été détaillée dans le chapitre précédent. Dans un premier temps, nous proposons une classification des caractéristiques fonctionnelles des SMA. Cette classification a pour but d'orienter notre choix des critères à évaluer durant l'étape suivante. Nous présentons ensuite les critères choisis ainsi que le modèle adopté pour caractériser ces derniers : les graphes. En se basant sur la théorie des graphes nous proposons des mesures pour l'évaluation des SMA et nous proposons une synthèse des interprétations possibles de ces mesures.

1. Critères d'évaluation des SMA

Comme expliqué dans la section 1 du chapitre 2, pour évaluer les SMA nous nous basons sur l'analyse de leurs caractéristiques fonctionnelles. Ces différentes caractéristiques couvrent diverses facettes des SMA, pour cette raison nous en proposons d'abord une classification. L'identification des catégories vise à mieux orienter nos choix quant aux critères d'évaluation adoptés et mesurés.

1.1. Classification des critères d'évaluation

Boissier et al. [Boi04] énumèrent et expliquent les caractéristiques fonctionnelles des SMA de manière volontairement linéaire sans classification ni structuration préétablies. Cependant, une lecture approfondie de ces caractéristiques nous a permis de dégager des traits communs à ces dernières et de les scinder en plusieurs catégories. Nous proposons leur classification en trois catégories :

- **Propriétés structurelles** : Cette catégorie caractérise la manière dont les agents sont organisés et les relations qui existent entre eux. Elle regroupe les caractéristiques suivantes : la communication, l'interaction, l'organisation, la distribution et la décentralisation.

- **Propriétés comportementales** : cette catégorie caractérise la manière dont le SMA se comporte vis-à-vis de lui-même, de ses composants, de son environnement et des autres systèmes éventuels auxquels il serait relié. Elle permet de décrire également la manière dont il évolue notamment dans le temps. Elle regroupe les caractéristiques suivantes : l'autonomie, l'ouverture, l'adaptation et l'émergence.

– **Propriétés d'interfaçage** : cette catégorie caractérise la manière dont le SMA interagit avec l'extérieur que ce soit son environnement, un autre système ou un acteur humain. Elle regroupe les caractéristiques suivantes : la situation dans un environnement, la délégation, l'intelligibilité et la personnalisation.

En adoptant cette vision des caractéristiques des SMA, il nous est désormais plus facile d'orienter nos choix en nous focalisant sur une catégorie particulière de propriétés.

1.2. Choix des critères d'évaluation

Les diverses études et revues de la littérature dans le domaine des SMA mettent toujours en évidence la démarcation de ces derniers en termes de dynamique et de socialité. En effet, un SMA est un système distribué au sein duquel les agents s'exécutent de manière parallèle et interagissent à travers des échanges asynchrones d'informations. Les interactions entre agents jouent ainsi un rôle déterminant dans la dynamique du système. Au-delà de cette dynamique, ces interactions traduisent des situations de dépendance et font apparaitre une vraie dimension sociale. Les différentes situations d'interactions entre agents dans un SMA dévoilent des structures sociales et organisationnelles qu'il convient d'étudier.

Dans cette thèse, nous avons donc choisi d'aborder l'analyse de l'aspect structurel des SMA à travers l'évaluation des caractéristiques de communication et d'organisation. Le choix de la communication en premier lieu se justifie par le fait qu'elle soit la brique de base et le pilier permettant de mettre en œuvre concrètement la panoplie d'interactions possibles sans un SMA. Quant au choix de l'organisation, il est justifié par l'importance de la caractérisation des structures sociales qui peuvent apparaitre dans un SMA.

Pour caractériser ces deux dimensions dans le cadre de l'approche globale d'évaluation proposée (se résumant aux étapes d'observation, modélisation et mesure) nous avons maintenant besoin d'un moyen permettant d'exprimer et de représenter l'aspect structurel d'un SMA, c'est la question que nous abordons dans la section suivante.

2. Modélisation par les graphes

D'une manière générale, la structure d'un système est déterminée par la définition des entités qui le composent et des relations qui existent entre ces entités. Il en va de même pour un SMA, la compréhension de sa structure repose sur la détermination des agents qui le composent et de leurs interrelations. Ces dernières sont elles-mêmes exprimées

grâce aux différentes interactions dans lesquelles les agents s'engagent et qui se manifestent concrètement à travers la communication. La tâche à laquelle nous sommes confrontés à ce niveau consiste à déterminer le formalisme adéquat qui permette de modéliser un SMA du point de vue de sa structure.

Il convient de préciser dans ce cadre le niveau de granularité de la représentation désirée. La revue de l'état de l'art dans le domaine de la simulation multi-agents nous a permis de dégager trois principaux niveaux de granularité selon lesquels les SMA peuvent être représentés [Lep00] [Gau07] [Mei07] [Lam11] [Nav12] :

- **Niveau microscopique** : il s'agit d'un niveau de granularité très fin, selon lequel tous les détails relatifs aux agents, à leurs comportements, à leurs architectures internes et à leurs interactions sont représentés. Ce niveau de granularité pose deux principaux problèmes. Le premier concerne l'importance de la quantité d'information générée dans ce type de représentation. Une quantité qui s'accroît d'autant plus avec le nombre d'agents et la complexité de leurs états et de leurs comportements. Le second problème découle du premier et pose la question de la gestion et de la manipulation d'une telle représentation.

- **Niveau macroscopique** : il s'agit d'un niveau de granularité grossier, selon lequel un SMA est considéré au pire des cas comme une entité singulière et au meilleur des cas comme une agrégation de groupes d'agents. Ce niveau de granularité permet de réduire considérablement la complexité de la représentation générée et par conséquent de faciliter sa gestion et son analyse. Cependant, les simplifications faites dans ce cadre sont souvent au détriment de la fidélité et de la précision de l'adéquation du modèle représentatif au modèle réel.

- **Niveau mésoscopique** : il s'agit d'un niveau de granularité intermédiaire qui se place entre le niveau microscopique et le niveau macroscopique, selon lequel un SMA est vu comme une agrégation d'agents en interaction. Ces derniers étant considérés comme des entités atomiques, dont les états et les comportements internes ne sont pas représentés. Le niveau mésoscopique offre un bon compromis entre la complexité et la précision de la représentation du SMA.

Dans notre démarche, l'architecture interne des agents nous importe peu, en nous focalisant sur la structure du SMA, nous nous plaçons à niveau mésoscopique. En effet nous nous affranchissons des états et des comportements internes des agents. Ces derniers

sont donc considérés comme des entités atomiques interconnectées. Le formalisme de modélisation adopté doit donc se concentrer principalement sur ces entités et leurs interconnections.

Suite à la définition de nos besoins de modélisation, c'est donc d'une manière tout à fait naturelle et intuitive que nous nous sommes orientés vers les graphes comme formalisme de représentation. En effet, les graphes permettent précisément de représenter les relations entre les entités d'un système. Il s'agit d'un moyen de modélisation très puissant dont les capacités et les possibilités de représentation qu'il offre sont incontestables. C'est pour cette raison que les graphes se voient utilisés dans divers domaines d'applications qui couvrent aussi bien les sciences techniques (biologie, chimie, physique, informatique) que les sciences humaines (géographie, histoire, sociologie) ou encore les sciences économiques (finance, logistique). En informatique, les graphes permettent, entre autres, de représenter divers types de réseaux et offrent un moyen d'abstraction et de modélisation des entités réelles composant ces réseaux. L'analyse des graphes s'appuie sur la théorie des graphes qui est une branche des mathématiques s'intéressant à leur encodage ainsi qu'à l'analyse de leurs propriétés.

Dans le contexte de notre thèse, notre solution consiste, dans un premier lieu, à modéliser le SMA par un graphe orienté dont les nœuds représentent les agents et les arcs représentent les interactions entre ces agents. Nous proposons, dans un second lieu, de dégager les propriétés de ce graphe en se basant sur la théorie des graphes, de les étudier et de les exploiter pour évaluer les caractéristiques de communication et d'organisation dans les SMA. Dans ce qui suit, nous définissons de manière plus formelle le modèle adopté.

2.1. Modèle de graphe statique

D'après [Gon95] un graphe $G = [X,U]$ orienté est déterminé par la donnée de :

- Un ensemble X de sommets. Si $|X| = N$, on dit que le graphe G est d'ordre N.
- Un ensemble U de couples ordonnés de sommets appelés arcs. On notera souvent $|U| = M$.

Ici les sommets représentent les agents du SMA et les arcs représentent les liens de communication entre ces agents. Un arc étant pondéré par le nombre de messages échangés entre les sommets qu'il relie.

Pour décrire un graphe, diverses représentations possibles peuvent être utilisées. Nous proposons d'utiliser la *matrice d'adjacence* appelée aussi *matrice d'incidence sommets-sommets*. C'est une matrice à coefficients 0 ou 1 :

$$A = \left(A_{ij}\right)_{\substack{i=1..N \\ j=1..N}}$$

Où chaque ligne (respectivement colonne) correspond à un sommet de G et où :

$$A_{ij} = 1 \quad si\ et\ seulement\ si \quad (i,j) \in U \quad \left(A_{ij} = 0\ sinon\right).$$

Dans le but de raffiner cette modélisation, nous proposons de pondérer chaque arc par le nombre de messages y étant passés. Nous utilisons donc une autre matrice *P* que nous appelons *matrice des poids* pour sauvegarder les poids des différents arcs :

$$P_{ij} = p(u) \quad si\ et\ seulement\ si \quad u = (i,j) \in U \quad \left(P_{ij} = 0\ sinon\right).$$

2.2. Modèle de graphe dynamique

Les SMA sont des systèmes caractérisés essentiellement par leur forte évolution dans le temps, un SMA peut être constamment soumis à des changements dans la mesure où l'on peut observer l'ajout ou la suppression d'agents, la mise en place de nouvelles relations entre les agents, etc. Ces évènements apportent des changements au niveau de la structure du graphe et par conséquent sur ses caractéristiques. Toutes ces modifications doivent être représentées par le modèle proposé qui doit être le plus fidèle possible.

Nous proposons donc d'étendre le modèle décrit précédemment en un graphe dynamique D'un point de vue pratique, un nœud est un objet qui possède un identifiant et un arc est un objet qui possède trois attributs :

- l'identifiant du nœud source
- l'identifiant du nœud destination
- un poids : vecteur

Le poids d'un arc est un vecteur de marquage à deux composantes :

- le temps
- le poids effectif (nombre de messages échangés)

Il convient de répertorier l'état du graphe à chaque modification de ce dernier, de manière à être en mesure de retrouver ses états précédents lors de l'étape de mesures de performances. La génération de la trace du graphe se fera de manière évènementielle,

c'est-à-dire qu'elle sera déclenchée à chaque fois qu'un évènement significatif qui induit le changement de la structure du graphe aura lieu.

3. Mesures de performances

Une fois que nous avons une modélisation de notre système multi-agents par un graphe, nous pouvons exploiter les propriétés de ce graphe dans l'étape d'évaluation.

3.1. Évaluation de la communication

Cette section est consacrée à l'étude de la communication dans les SMA et à la proposition de mesures adéquates pour son évaluation en se basant sur la théorie de graphes.

3.1.1. La communication dans les SMA

La communication a toujours été considérée comme une notion de grande importance en informatique. Particulièrement prisée dans les systèmes répartis, elle présente un intérêt tout aussi considérable dans les SMA. Cependant, la communication entre les entités d'un système multi-agents diffère de la communication entre les processus d'un système distribué classique dans la mesure où elle s'effectue à un niveau d'abstraction plus élevé.

La communication est une composante centrale à la base des interactions et au cœur de l'organisation sociale dans les SMA [Cha01]. Elle permet aux agents d'échanger des informations afin de coopérer, de coordonner leurs actions et de réaliser un objectif commun, devenant ainsi de véritables êtres sociaux [Fer95].

Il existe trois principaux modes de communication dans les SMA :

- La communication par émission de signal dans l'environnement : ce mode est caractéristique des SMA réactifs où les agents évoluent dans l'environnement et y laissent des traces sous formes de signaux détectables par les autres agents.
- La communication par partage d'information sur un support commun : dans ce cas, les agents communiquent intentionnellement en rendant visibles par les autres agents les informations qu'ils désirent transmettre. Il s'agit du mécanisme de tableau noir.
- La communication directe par échange de messages : dans ce cas, les agents échangent des informations en s'envoyant des messages à travers un medium d'acheminement.

Dans cette thèse, nous nous intéressons particulièrement au mode de communication directe vu que c'est le plus simple et le plus répandu.

3.1.2. Mesures proposées pour l'évaluation de la communication

Pour appréhender la structure de notre graphe de communication, nous pouvons commencer par la caractériser localement en regardant pour chaque sommet (agent) les autres sommets (agents) auxquels il est relié. Nous déterminons pour chaque sommet *n* :

- **Le demi-degré extérieur $d^+(n)$**

C'est le nombre d'arcs ayant le nœud *n* comme source. Il reflète le degré de participation de l'agent correspondant à la communication. Si nous utilisons la représentation du graphe par sa matrice d'adjacence, nous aurons alors :

$$d^+(n) = \sum_{j=1}^{N} A_{nj}$$

Soit l'exemple du graphe illustré par la figure 3.1 ci-après.

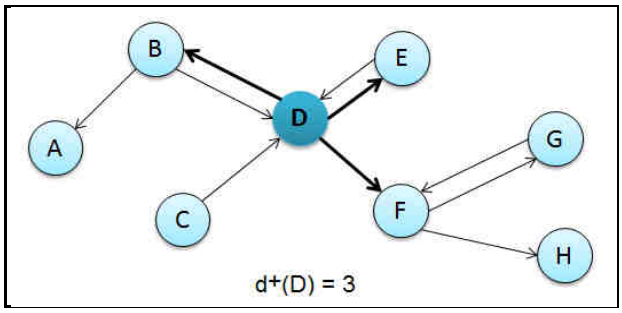

Figure 3.1 : Exemple de calcul du demi-degré extérieur

Considérons le cas du nœud D, ce dernier possède 3 arcs sortants donc son demi-degré extérieur $d^+(D)$ est égal à 3.

- **Le demi-degré intérieur $d^-(n)$**

C'est le nombre d'arcs ayant *n* comme destination. Il reflète le degré de sollicitation de l'agent correspondant à l'acte de communication. Si nous utilisons la représentation du graphe par sa matrice d'adjacence, nous aurons alors :

$$d^-(n) = \sum_{i=1}^{N} A_{in}$$

Soit l'exemple du même graphe précédent illustré par la figure 3.2 ci-dessous.

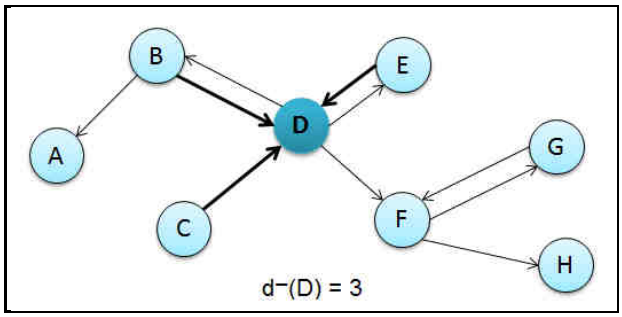

Figure 3.2 : Exemple de calcul du demi-degré intérieur

Considérons le cas du même nœud D, ce dernier possède 3 arcs entrants donc son demi-degré intérieur $d^-(D)$ est égal à 3.

Ils existent d'autres grandeurs qui servent à décrire les propriétés structurelles d'un graphe : *les indices*. Plusieurs indices peuvent être utilisés pour analyser les performances d'un réseau de communication. Ces indices ont été développés par Kansky en 1963 dans le but d'évaluer les réseaux géographiques de transports [Kan63]. Nous nous contentons ici de citer ceux qui s'avèrent utiles pour notre évaluation.

- **L'indice bêta β**

Il indique la relation entre le nombre d'arcs (*M*) sur le nombre de sommets (*N*). Plus β est élevé, plus le réseau de communication est complexe. Cet indice reflète donc, la complexité de la structure du réseau de communication qui relie les différents agents du SMA.

$$\beta = \frac{M}{N}$$

Dans l'exemple de graphe illustré par la figure 3.3 ci-dessous, le nombre de nœuds est égal à 8 et le nombre d'arcs est égal à 10, ce qui donne un indice β = 1.25.

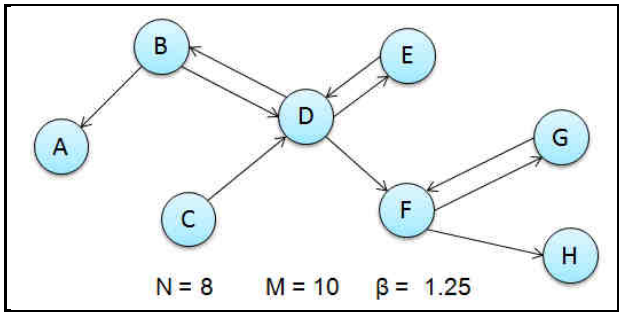

Figure 3.3 : Exemple de calcul de l'indice β

- **L'indice gamma γ**

Il indique la relation entre le nombre de liens observés et le nombre de liens possibles. Sa valeur est toujours comprise entre 0 et 1. Une valeur de 1 indique un réseau complètement connecté. Cet indice est très efficace pour évaluer la progression d'un réseau dans le temps. Dans notre cas, il permet de mesurer le degré de communication dans le SMA.

$$\gamma = \frac{E}{\frac{1}{2} N(N-1)}$$

pour un graphe non orienté et

$$\gamma = \frac{E}{N(N-1)}$$

pour un graphe orienté, avec

$$E = \sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij}$$

Dans l'exemple du graphe illustré par la figure 3.4 ci-dessous, on observe 6 liens alors que le nombre de liens possibles est égal à 12. L'indice γ est donc égal à 0.5.

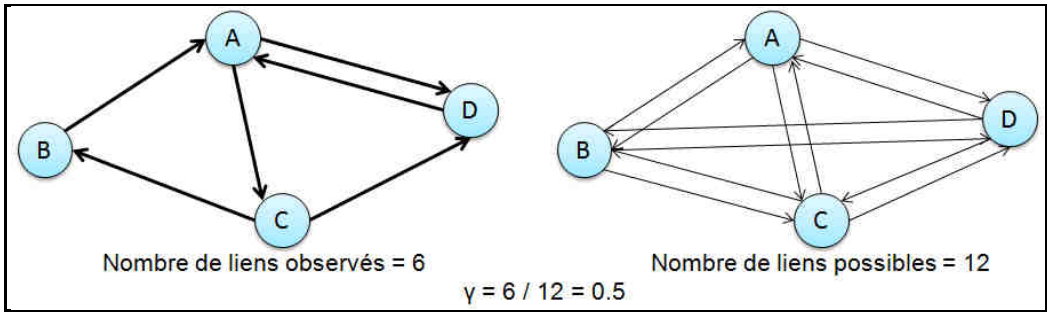

Figure 3.4 : Exemple de calcul de l'indice γ pour un graphe orienté

- **L'indice thêta θ**

Il mesure la quantité de trafic moyen par sommet. Plus θ est élevé plus la charge du réseau est grande.

$$\theta = \frac{Q(G)}{N} \quad \text{avec} \quad Q(G) = \sum_{i=1}^{N} \sum_{j=1}^{N} P_{ij}$$

Dans l'exemple illustré par la figure 3.5 ci-après, la charge totale du réseau est égale à 17. Le nombre de nœud étant égal à 4, θ aura donc une valeur de 4.25.

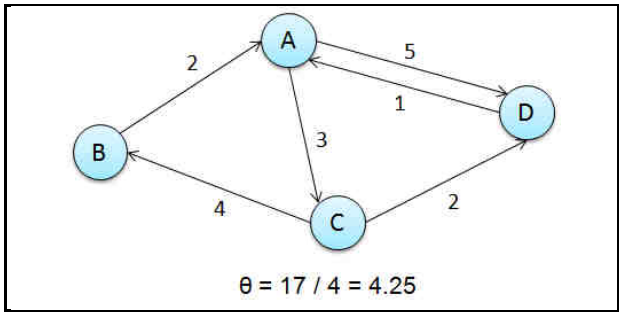

Figure 3.5 : Exemple de calcul de l'indice θ

Nous pouvons aussi calculer la charge *Q(n)* de chaque sommet et mesurer ainsi l'équité de la répartition des charges entre les différents agents.

$$Q(n) = \sum_{\substack{i=1 \\ i \neq n}}^{N} P_{in} + P_{ni}$$

Dans la figure 3.6 ci-dessous, sont présentées les différentes valeurs des charges des nœuds du graphe adopté pour exemple.

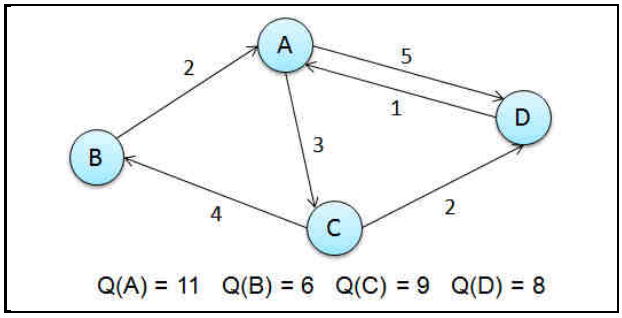

Figure 3.6 : Exemple de calcul de la charge de chaque nœud d'un graphe

- **La connexité**

Un graphe est dit connexe lorsque, pour tout couple de sommets, il existe une chaîne les joignant. L'étude de la connexité du graphe de communication nous renseigne sur l'organisation des agents du SMA. En effet, le fait de dégager les composantes connexes du graphe revient à identifier les différents réseaux d'accointances du SMA (c'est-à-dire les groupes d'agents communicant et appartenant à un même voisinage). Nous proposons, à cet effet, d'appliquer l'algorithme de Tarjan [Gon95] pour la recherche des

composantes connexes dans un graphe. Dans la figure 3.7 suivante sont illustrés deux types de graphes, le premier à gauche est un graphe connexe et le second à droite est un graphe non connexe ayant deux composantes connexes.

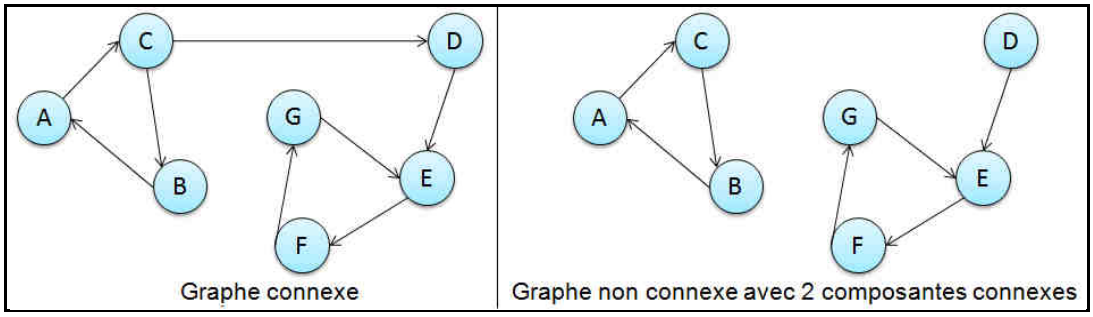

Figure 3.7 : Exemples de graphes avec différentes connexités

- **Les points d'articulation**

Un point d'articulation dans un graphe est un sommet dont la suppression augmente le nombre de composantes connexes. La recherche des points d'articulation dans le graphe de communication est une question assez importante. En effet, l'existence de tels points dans le réseau de communication du SMA reflète une certaine centralisation au niveau de la communication car le retrait de l'agent correspondant peut rendre isolé deux groupes d'agents et affecter le fonctionnement du SMA. Ainsi, moins il y a de points d'articulation mieux c'est. Nous proposons d'utiliser l'algorithme de Tarjan [Gon95] pour la recherche des points d'articulation dans une composante connexe dans un graphe.

La figure 3.8 ci-dessous illustre un exemple de graphe avec un point d'articulation (figure à gauche). Le nœud D est un point d'articulation car sa suppression entraine l'augmentation du nombre de composantes connexes : on note dans cet exemple le passage d'une seule composante connexe à deux composantes connexes (figure à droite).

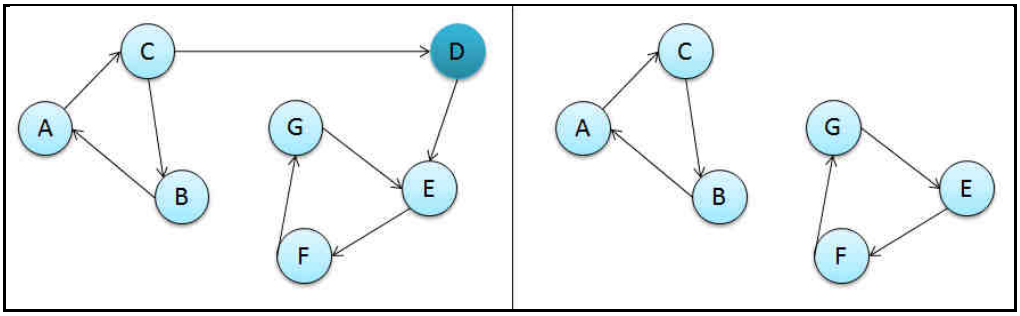

Figure 3.8 : Exemple de graphe avec un point d'articulation

Au-delà des propriétés structurelles de la communication, nous nous intéressons également à l'évaluation de l'aspect syntaxique de cette dernière. Cet aspect ne s'appuie pas forcément sur le modèle de graphe dans son analyse mais s'avère d'une grande importance dans la mesure où il permet de toucher le fond des messages échangés. L'aspect syntaxique de la communication est étudié à travers l'évaluation de la richesse de la typologie des messages échangés ainsi que de la complexité de leurs contenus.

- **Typologie des messages**

La communication entre agents repose sur la théorie des actes du langage qui suppose que les verbes ont les caractéristiques des actions dans le sens où ils peuvent changer l'état du monde de manière analogue aux actions physiques. Le philosophe John Austin (1962) a identifié un certain nombre de verbes performatifs qui correspondent à des différents types d'actes du langage. On cite à titre d'exemples les verbes *demander*, *informer*, *promettre*, *répondre*, *affirmer*, *dénier*, etc. Les langages de communication entre agents se basent tous sur la théorie des actes du langage et définissent une liste de performatives correspondant aux différents actes de communication accomplis. Nous proposons d'étudier la richesse de la typologie des messages échangés dans le SMA. Pour ce faire, nous comptons extraire le champ performative de chaque message et faire une étude statistique sur le nombre total de performatives utilisées dans le SMA.

- **Complexité des messages**

Selon la structuration du contenu des messages, nous comptons attribuer à chaque message une parmi ces trois qualifications {*simple*, *moyen*, *complexe*}. Cette qualification diffère selon que le contenu du message se présente sous l'une des formes suivantes :

- un texte brut et dans ce cas il s'agit d'un contenu *simple* étant donné qu'il ne nécessite pas d'être encodé et décodé pour être interprété.
- une proposition basée sur une ontologie et dans ce cas il s'agit d'un contenu *moyennement complexe* étant donné qu'il est encodé sous une forme propre au SMA et nécessite donc d'être décodé conformément à l'ontologie commune pour être interprété par les agents.
- un message conforme à un protocole bien déterminé et dans ce cas il s'agit d'un contenu *complexe* étant donné que mise à part l'utilisation d'une ontologie commune, le protocole spécifie aussi un certain nombre d'actions et de comportements bien déterminés.

Nous complétons notre étude des aspects structurel et syntaxique de la communication par quelques mesures qui permettent de dimensionner le SMA étudié. Il s'agit notamment des mesures suivantes :

- Le nombre de messages échangés ;
- La taille de ces différents messages ;
- Le nombre d'agents impliqués dans la communication ;
- Le pourcentage de ces agents par rapport au nombre total des agents du SMA.

3.2. Évaluation de l'organisation

Cette section est consacrée à l'étude de l'organisation dans les SMA et à la proposition de mesures adéquates pour son évaluation en se basant sur la théorie de graphes.

3.2.1. L'organisation dans les SMA

Un SMA est une société d'agents en interaction mutuelle, se partageant les tâches, les ressources et les connaissances. Il y a un large panel d'interactions possibles entre les agents qui comprend la coordination, la collaboration, la négociation, etc. [Fer95].

Les différents schémas d'interaction dans un SMA engendrent des structures organisationnelles et des réseaux d'accointances inter-reliés. Les organisations dans les SMA peuvent se former de deux manières :

- Elles peuvent être formées a priori par le concepteur du système et dans ce cas on parle d'organisation prédéfinies ;
- Elles peuvent être formées a posteriori comme résultat des comportements des différents agents et de leurs interactions et dans ce cas on parle d'organisations émergeantes [Fer95].

Au-delà de la manière dont elles peuvent se former, les organisations dans les SMA sont caractérisées par la nature des relations qui existent entre leurs entités. Ces relations peuvent être égalitaires, et dans ce cas les agents participent équitablement à la prise de décision. On peut également distinguer des relations de subordination ou d'autorité qui permettent de définir des structures organisationnelles hiérarchiques.

Dans [Hor05] Horling et Lesser identifient neuf différentes structures organisationnelles possibles au sein des SMA et en présentent une étude très intéressante. Les styles d'organisations qu'ils évoquent sont les suivants :

- Les hiérarchies : les organisations hiérarchiques impliquent l'existence de relations de subordination entre agents. La prise de décision est la responsabilité d'un agent supérieur. Lorsqu'un tel agent est unique on parle de hiérarchie simple. Dans le cas où l'autorité est distribuée sur plusieurs entités on parle de hiérarchie uniforme.

- Les holarchies : la dénomination holarchie fait référence au terme holon, il s'agit d'un groupe d'agents lui-même composé de plusieurs groupes d'agents d'un niveau inférieur au sens d'une hiérarchie il s'agit donc d'un type particulier de hiérarchies.

- Les coalitions : ce sont des organisations d'agents dirigées par les buts où les agents s'assemblent promptement et collaborent afin de réaliser leurs propres intérêts en l'absence d'un objectif global. Dans les coalitions, il n'existe pas de relations hiérarchiques explicites, cependant il peut y avoir un agent particulier désigné par les autres agents pour représenter la coalition.

- Les équipes : dans ce genre d'organisations, les agents s'assemblent dès le départ afin d'assurer la complémentarité des compétences dans la réalisation de l'objectif de l'équipe mais pas nécessairement leurs objectifs individuels.

- Les congrégations : elles présentent des similarités avec les coalitions et les équipes. Il s'agit d'organisations plates (non hiérarchiques) mais qui ne sont pas dirigées par les buts. Les agents se regroupent en fonction de la similarité de leurs compétences afin de faciliter la recherche de collaborations.

- Les sociétés : ce sont des organisations ouvertes dans la mesure où des agents de différentes catégories peuvent y entrer et en sortir au besoin. Les agents apparaissent et disparaissent mais la société persiste et agit comme un environnement au sein duquel les agents évoluent.

- Les fédérations : la formation de fédérations repose sur la délégation de l'autorité de la part d'un groupe d'agents à un agent délégué pour représenter le groupe. Les agents du groupe interagissent uniquement avec cet agent qui agit comme intermédiaire entre ce dernier et le monde extérieur.

- Les marchés : ce sont des organisations qui se basent sur le schéma producteur-consommateur. Les agents sont de deux catégories, ceux qui fournissent des services, gèrent des ressources ou assurent des tâches et ceux qui en font la demande, le tout dans un contexte caractérisé par la compétition.

– Les matrices : il s'agit d'organisations hiérarchiques dans lesquelles il n'existe pas d'unique supérieur pour chaque agent. Un agent peut être subordonné à plusieurs supérieurs hiérarchiques.

Au-delà de cette riche diversité des types d'organisations, ces dernières peuvent être composées et regrouper plusieurs structures organisationnelles différentes à la fois.

Ainsi, l'organisation est une caractéristique importante et un concept de base des SMA car elle exprime la dimension sociale de tels systèmes. L'évaluation de l'organisation est d'un grand intérêt dans la mesure où elle permet de divulguer et de comprendre la manière dont les SMA sont structurés pour la résolution de problèmes.

Dans [Fer95], Ferber définit les différentes manières possibles d'analyser l'organisation dans les SMA. D'une part il y a l'analyse fonctionnelle qui tend à identifier et décrire les fonctions que les différentes entités de l'organisation sont supposées réaliser. Ainsi l'organisation est vue comme un ensemble de rôles et de relations entre ces rôles. D'une autre part, il y a l'analyse structurelle qui se concentre sur les interactions entre les agents afin d'appréhender leurs interrelations complexes et d'expliquer les structures organisationnelles résultantes.

Dans nos travaux, nous nous intéressons précisément à l'aspect structurel des organisations. C'est pour cette raison que le modèle représentatif des SMA choisi se base sur les graphes. Les graphes permettent d'exprimer explicitement la structure de l'organisation en fournissant une image des différentes entités et leurs relations. En se basant sur ce modèle de graphe, l'évaluation de l'organisation est réalisée en se basant sur des mesures de la théorie des graphes.

3.2.2. Mesures proposées pour l'évaluation de l'organisation

Les mesures suivantes sont majoritairement basées sur les degrés des nœuds et pour la plupart issues des domaines de la théorie de l'organisation computationnelle [Kra94] et de l'analyse des réseaux sociaux [Bar06] [Mal09] vus leurs proximités avec les organisations d'agents.

- **Distribution des degrés**

Dans la théorie des graphes, le degré d'un nœud est le nombre de connexion que ce nœud possède avec les autres nœuds. Nous désignons par *Ki* le degré d'un nœud *i*.

La figure 3.9 ci-après illustre l'exemple d'un graphe et présente les valeurs des degrés des différents nœuds qui le composent.

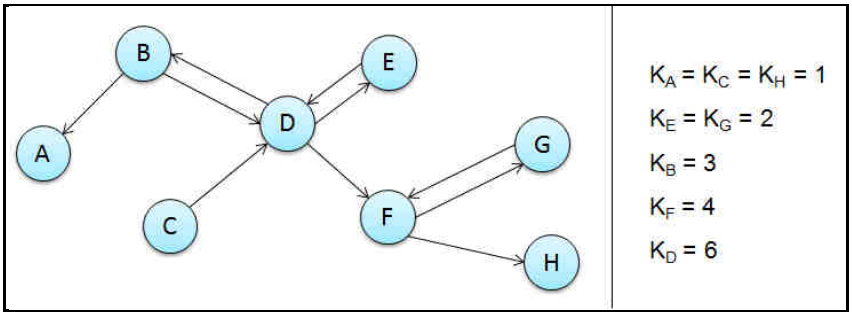

Figure 3.9 : Exemple de calcul des degrés des nœuds d'un graphe

Comme on peut le constater à travers l'exemple précédent, dans un graphe, les nœuds peuvent avoir différents degrés, ainsi la distribution des degrés *P(K)* est définie comme étant la proportion des nœuds dans le graphe ayant un degré égal à *K*.

$$P(K) = \frac{N_K}{N}$$

Avec *Nk* le nombre de nœuds ayant le degré *K* et *N* le nombre total de nœuds.

La figure 3.10 suivante illustre un exemple de graphe dont les nœuds possèdent différents degrés et présente les valeurs des distributions de ces degrés.

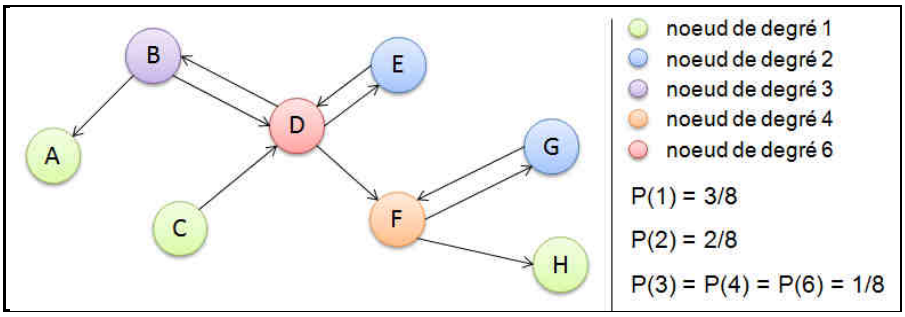

Figure 3.10 : Exemple de calcul de la distribution des degrés dans un graphe

La distribution des degrés est très importante dans l'analyse de la structure du graphe et permet d'appréhender ses propriétés topologiques. Les différents types de réseaux possèdent différentes distributions de degrés *P(K)*. Ainsi l'allure de la fonction de distribution des degrés permet de distinguer deux grandes classes de réseaux : homogènes et hétérogènes (voir Figure 3.11)

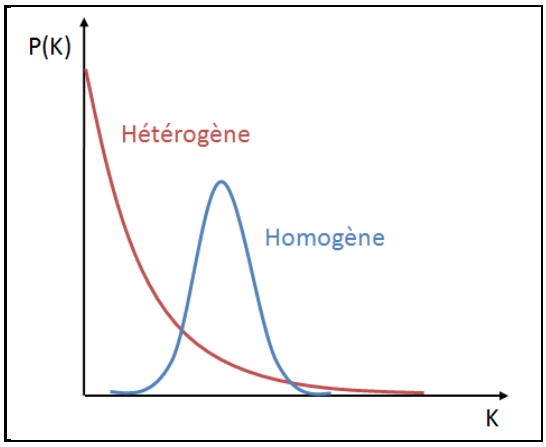

Figure 3.11 : Distribution des degrés homogène vs. hétérogène

Les réseaux homogènes sont caractérisés par une distribution de degrés concentrée autour de sa valeur moyenne $< K >$ donnée par :

$$< K > = \sum K \times P(K)$$

Dans de tels réseaux, il n'y a pas de nœuds ayant de grands degrés par rapport aux autres. Les réseaux hétérogènes sont, quand à eux, caractérisés par une distribution de degrés qui généralement suit une loi de puissance. Ceci est le cas de la plupart des réseaux du monde réel. Cela signifie que dans ces réseaux il a une large majorité de nœuds ayant un degré faible et un petit nombre d'agents ayant un haut degré qui dépasse largement la moyenne. Ces derniers sont considérés comme ayant des fonctions spéciales dans leur réseau.

- **Le degré de voisinage**

Le degré de voisinage $K_{nn,i}$ exprime la moyenne des degrés des voisins d'un nœud *i*. Il s'agit d'un moyen simple de caractériser le type d'environnement dans lequel le nœud se situe. Le degré de voisinage d'un nœud est déterminé de la manière suivante :

$$K_{nn,i} = \frac{1}{K_i} \sum_j K_j$$

Dans la figure 3.12 suivante, sont calculés et présentés les degrés de voisinage des différents nœuds du graphe adopté pour exemple.

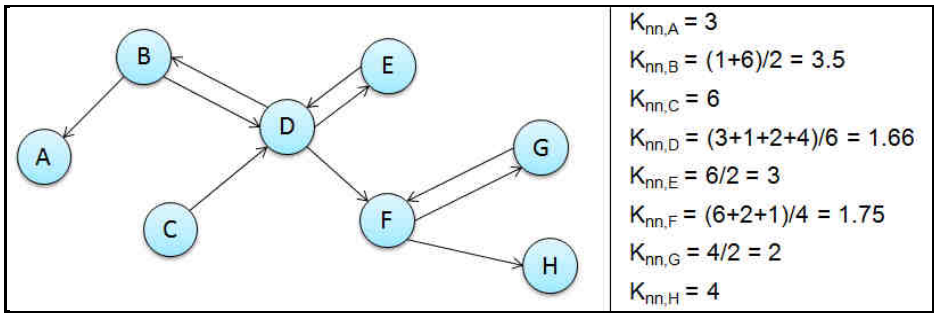

Figure 3.12 : Exemple de calcul des degrés de voisinage des nœuds d'un graphe

- **La corrélation-degré (assortativité)**

L'assortativité est une caractéristique des graphes qui reflète la tendance des nœuds à être plutôt en relation avec des nœuds de degrés similaires. Elle est souvent évaluée en termes de corrélation entre les degrés des nœuds. En se basant sur la mesure des degrés de voisinage précédemment définis, le corrélation-degré est donné par la moyenne des degrés de voisinage des nœuds de degré *K*.

$$K_{nn}(K) = \frac{1}{N_K} \sum_{i/K_i=K} K_{nn,i}$$

Considérons le même exemple de graphe précédemment utilisé, la figure 3.13 suivante présente les mesures des corrélation-degrés des différents degrés associés aux nœuds de ce graphe. Par exemple, dans ce graphe, il y a trois nœuds ayant un degré égal à 1, qui sont A, C et H. Les degrés de voisinage respectifs de ces nœuds sont 3, 6 et 4. $K_{nn}(1)$, étant le degré moyen des voisins des nœuds de degré 1, est égal à (3+6+4)/3.

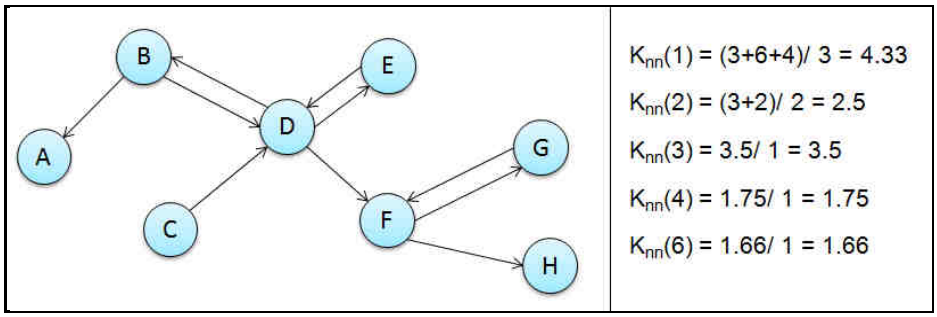

Figure 3.13 : Exemple de calcul des corrélation-degrés d'un graphe

Le comportement $K_{nn}(K)$ définit deux classes principales de réseaux. Nous parlons de réseaux assortatifs lorsque la valeur de $K_{nn}(K)$ accroit en fonction de *K*. C'est le cas des réseaux sociaux dans lesquels les nœuds de haut degré sont préférentiellement associés

avec des nœuds de degré similaires. Et nous parlons de réseaux disassortatifs lorsque la valeur de $K_{nn}(K)$ décroit en fonction de K. C'est plutôt le cas des réseaux hiérarchiques dans lesquels les nœuds de haut degré tendent à se connecter avec plusieurs nœuds de degré plus bas.

- **La centralité**

La centralité est une caractéristique exprimant la position d'un nœud dans le graphe. Elle permet de déterminer l'importance d'un nœud dans le réseau et à quel point il est influent. Ce concept s'inspire de l'analyse des réseaux sociaux. Plusieurs indicateurs sont utilisés pour mesurer la centralité, ici nous utilisons le degré de centralité. *Cd(i)* désigne le degré de centralité du nœud *i*.

$$Cd(i) = \frac{Ki}{(N-1)}$$

Avec *Ki* le degré du nœud *i* et *N* le nombre de nœuds.

Dans la figure 3.14 ci-dessous, nous présentons les mesures des degrés de centralité de chacun des nœuds composant le graphe donnée en exemple.

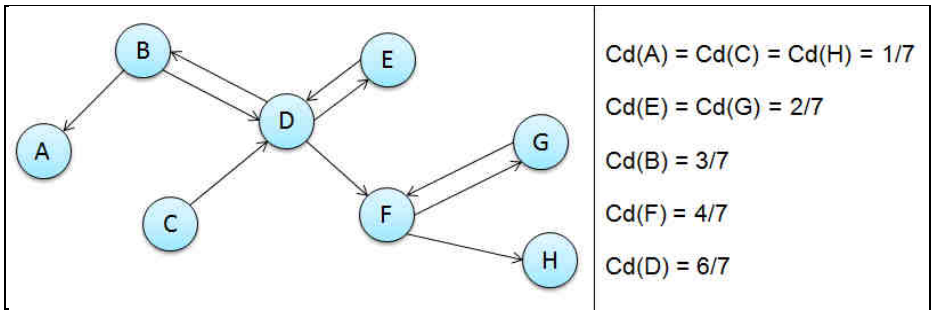

Figure 3.14 : Exemple de calcul des degrés de centralité des nœuds d'un graphe

- **La hiérarchie**

Nous parlons de hiérarchie dans les organisations si les relations entre les différentes entités sont déterminées par l'autorité. Dans les graphes orientés, la hiérarchie est définie somme suit :

$$Hierarchy = 1 - \left[\frac{V}{MaxV}\right]$$

Avec *V* le nombre de paires de nœuds symétriquement connectés c'est-à-dire les nœuds qui peuvent mutuellement s'atteindre, et *MaxV* le nombre total de paires de nœuds [Kra94].

Cette mesure est intéressante pour exprimer la hiérarchie d'un réseau mais peut dans certains cas ne pas très bien refléter degré de hiérarchie dans les SMA car elle ne prend pas en considération la sémantique du lien entre deux nœuds. En effet, le degré d'hiérarchie dans les SMA devrait mettre en évidence l'existence d'agents dominants et autoritaires. Pour cette raison, nous proposons d'examiner la nature des différents liens entre les agents afin de déterminer s'il existe ou non une relation de force. Le meilleur moyen pour ce faire consiste à relever la nature du message échangé afin d'identifier les messages d'ordre. Ainsi une mesure complémentaire proposée pour la hiérarchie est la suivante :

$$Hierarchie = \frac{OrdV}{MaxV}$$

Avec OrdV le nombre de paires de nœuds reliés par une relation d'autorité (existence de messages d'ordre).

- **Le leadership**

Comme expliqué précédemment, dans les SMA différents agents peuvent s'assembler en structures organisationnelles appelées groupes qui peuvent être des équipes ou des coalitions. Dans les deux cas, il pourrait y avoir un agent ayant la capacité spéciale de coordonner les actions des autres agents, cet agent est appelé « Leader » [Leg03]. Alors que dans les équipes le leader représente une autorité de décision, dans les coalitions le leader n'a pas de réel pouvoir décisionnel mais est plutôt considéré comme un point de centralisation de l'information.

Afin d'identifier ce type d'agents moyennant le formalisme des graphes, un groupe est assimilé à une structure en étoile, avec le leader dans le centre entouré d'un groupe d'agents [Leg03]. Un sous-graphe *Sn* en étoile d'ordre *n* est un graphe de *n* nœuds avec un nœud ayant le degré *n-1* et les autres nœuds ayant le degré 1 [Har94].

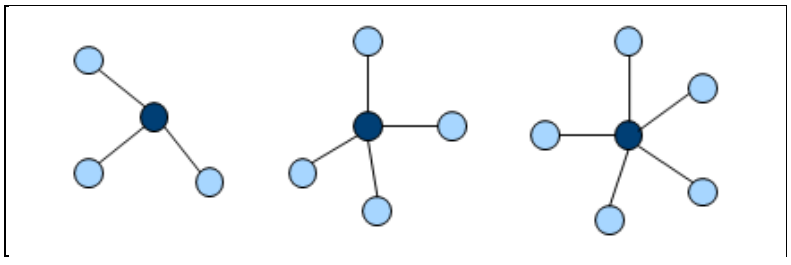

Figure 3.15 : Exemples de graphes en étoile

Ainsi, pour identifier les leaders dans un SMA nous devons identifier les sous graphes en étoile dans le graphe global représentatif du SMA.

3.3. Récapitulatif des mesures et interprétations possibles

Au terme de cette partie, nous nous proposons de dresser un résumé des différentes mesures adoptées pour l'évaluation des caractéristiques de communication et d'organisation. L'objectif de ce résumé consiste à fournir des interprétations possibles pour chacune de ces mesures indépendamment de l'application multi-agents évaluée.

De toute évidence, il est très difficile de dissocier une mesure de performance du contexte de son utilisation. En effet, l'interprétation d'une mesure est toujours liée au système en question et est spécifique à son fonctionnement. De plus une analyse réussie nécessite la connaissance profonde et la compréhension parfaite du système évalué [Jai91].

Cependant, dans le cadre de ce travail nous avons volontairement tenté, malgré cette difficulté, de dégager des interprétations génériques dans le but d'offrir à l'évaluateur un moyen supplémentaire d'aide et d'accompagnement dans sa prise de décision ultérieure.

Ainsi, à travers les tableaux 3.1 et 3.2 suivants nous résumons les différentes mesures associées respectivement à la communication et l'organisation dans les SMA. Pour chaque mesure nous déterminons s'il s'agit s'une mesure locale, c'est-à-dire permettant de caractériser un agent ou un lien de communication, ou bien d'une mesure globale c'est-à-dire permettant de caractériser tout le SMA. Nous présentons également les interprétations génériques possibles de ces différentes mesures sous forme d'indications qui seront développées plus tard en se basant sur les spécificités de chacune des applications multi-agents évaluées.

Mesures	Locale	Globale	Indications d'interprétations
Demi-degré extérieur $d^+(n)$	X		Participation de l'agent à la communication – Autonomie / proactivité (si messages de type information) – Dépendance (si messages de type requête)
Demi-degré intérieur $d^-(n)$	X		Sollicitation de l'agent à la communication – Détention de ressources – Détention de compétences
Indice bêta β		X	Complexité du réseau de communication – Agents cognitifs et parallélisme (si β haut) – Agents réactifs et parallélisme (si β bas)
Indice gamma γ		X	Connectivité du SMA
Indice thêta θ		X	Quantité de trafic moyen par agent Importance de la charge du réseau Interactivité du SMA
Charge Q(n)	X		Charge de communication par agent Equité de répartition de la charge
Nombre de composantes connexes		X	Réseaux d'accointances
Nombre de points d'articulation		X	Points de centralisation Points de vulnérabilité
Typologie des messages		X	Richesse de la typologie
Complexité des messages	X		Complexité du codage/décodage des messages Temps d'interprétation des messages
Nombre de messages		X	Abondance de la communication
Taille des messages	X		Temps d'acheminement et d'interprétation
Nombre d'agents communicants		X	Dimension du SMA
Pourcentage d'agents communicants		X	Degré de participation global à la communication

Tableau 3.1 : Récapitulatif et interprétations possibles des mesures de la communication

Mesures	Locale	Globale	Indications d'interprétations
Distribution des degrés P(K)		X	Nature des nœuds - Homogènes : Relations de parité, égal à égal (symétriques) - Hétérogènes : Relations d'autorité (asymétriques)
Degré de voisinage $K_{nn,i}$	X		Nature du voisinage direct et son (degré d'appartenance) - Cohésion forte avec le reste du SMA - Isolation
Corrélation-Degré K_{nn}		X	Nature du SMA - Assortatif - Disassortatif
Centralité C(d)	X		Dominance : - Contrôle des ressources - Détention des compétences - Prise de décision Agent délégué dans une : - Coalition - Fédération
Hiérarchie		X	Topologie de l'organisation - Hiérarchie (simple ou uniforme) - Holarchie - Fédération - Matrice
Leadership	X		Autorité : - Supériorité hiérarchique - Synchronisation - Arbitrage Topologie de l'organisation : fédération

Tableau 3.2 : Récapitulatif et interprétations possibles des mesures de l'organisation

Conclusion

Ce chapitre a été consacré à la proposition d'une approche d'évaluation des SMA en se basant sur la modélisation. Dans un premier temps nous avons classifié les différentes caractéristiques des SMA en trois catégories : structurelles, comportementales et d'interfaçage. Notre intérêt a porté essentiellement sur les propriétés structurelles, nous avons donc adopté la modélisation par graphes puisqu'ils correspondent parfaitement à la facette des SMA que nous souhaitons évaluer. Notre choix s'est orienté vers deux propriétés essentielles des SMA à savoir la communication et l'organisation, lesquelles ont été mesurées en se basant sur des métriques issues de la théorie des graphes. Dans le chapitre suivant nous mettons en œuvre les approches proposées ensuite nous exposons et interprétons les résultats obtenus.

CHAPITRE 4

EXPERIMENTATIONS ET RESULTATS

Sommaire

Après avoir défini les critères de mesure adoptés pour l'évaluation des caractéristiques structurelles des SMA, nous entamons à présent la phase de test et d'expérimentation. Dans ce quatrième chapitre, nous commençons par décrire notre démarche expérimentale à travers l'explication du fonctionnement du système d'évaluation proposé. Nous présentons par la suite les différents environnements de tests et de validation que nous avons utilisés. Il s'agit de deux applications multi-agents dont la première est une application de diagnostic des pannes dans un milieu industriel et la seconde est une application de gestion de la production dans les chaînes logistiques. Enfin les résultats et les interprétations des tests effectués sur ces dernières sont exposés.

1. Démarche expérimentale

Avant d'entamer les expérimentations et la présentation des résultats, nous commençons par décrire sommairement les moyens techniques utilisés pour mettre en œuvre le système d'évaluation proposé.

1.1. Techniques de mise en œuvre

Nous rappelons que le système d'évaluation se compose de trois modules assurant respectivement les tâches d'observation, de modélisation et de mesure. Le premier se compose essentiellement de sondes logicielles paramétrables définies grâce à la programmation orientée-aspects. Une sonde est donc un aspect qui consiste en un ensemble de points de coupure et de greffons. Les points de coupure définissent les évènements significatifs à intercepter et les greffons définissent les actions à effectuer lors de l'atteinte d'un point de coupure, notamment la génération des traces. Les sondes ont été développées avec AspectJ[1] une extension standard orientée-aspects pour le langage Java. Les traces d'exécution générées par les sondes sont sauvegardées dans un fichier XML.

Le module de modélisation s'intéresse à l'exploitation des ces traces, il procède à l'analyse syntaxique (parsing) du fichier généré afin de tracer le graphe de communication du SMA. Ce dernier est construit et affiché graphiquement en utilisant GraphStream[2] qui est une bibliothèque Java pour la manipulation et la représentation des graphes.

[1] http://eclipse.org/aspectj/

[2] http://graphstream-project.org/

Le module de mesure exploite ce graphe pour calculer les mesures de performance définies. Ces dernières sont présentées de manière graphique grâce à la bibliothèque JFreeChart[3]. Il s'agit d'une bibliothèque Java permettant de créer des graphiques et des diagrammes.

1.2. Scénario d'exécution

A travers la figure 4.1 ci-dessous, nous expliquons le principe de fonctionnement global du système d'évaluation proposé à travers un scénario d'exécution.

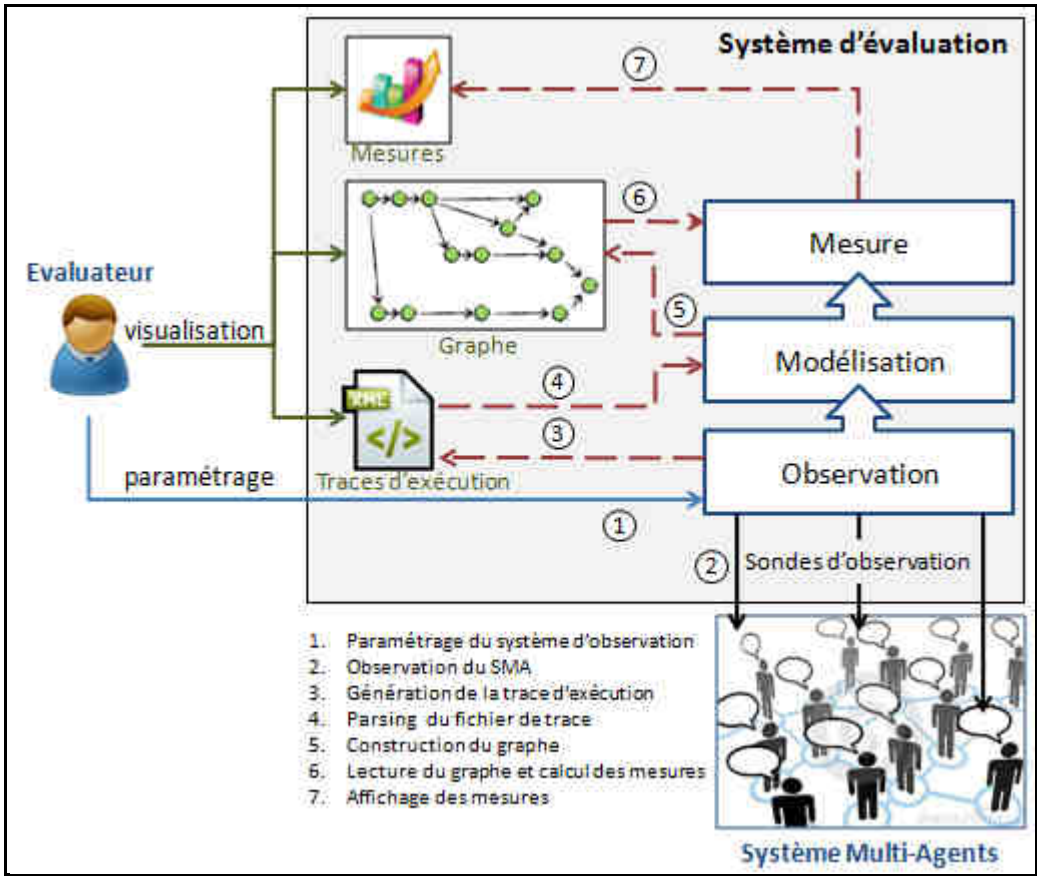

Figure 4.1 : Schéma descriptif du fonctionnement global du système d'évaluation

La démarche expérimentale se résume de la manière suivante. D'abord l'évaluateur est interrogé s'il souhaite lancer l'application multi-agents en mode normal ou en mode évaluation, auquel cas le module d'observation sera activé. L'évaluateur doit ensuite sélectionner la (les) caractéristique(s) à évaluer. Il s'agit de la première phase de fonctionnement des sondes d'observation qui est la phase de paramétrage. En effet, les

[3] http://www.jfree.org/jfreechart/

sondes se basent sur le choix de l'utilisateur pour définir les types d'évènements à observer. Dans notre étude, l'évaluation porte sur la communication et/ou l'organisation. Ainsi, les rubriques des sondes correspondant à l'observation et au traçage des échanges de messages entre agents seront activées lors de la phase de paramétrage.

Grâce à leur capacité d'interception du flot d'exécution de l'application muti-agents, les sondes captent chaque évènement relatif à un envoi ou une réception de message et déclenchent le greffon correspondant à la génération d'une trace. De cette manière, chaque évènement est répertorié à l'instant même de son occurrence, ce qui permet d'avoir une vue dynamique du comportement du système. L'utilisateur peut visualiser le rapport des évènements observés en temps réel.

Pour chaque évènement, sont sauvegardés son numéro, qui définit l'ordre dans lequel il s'est déroulé, son type et le temps de son occurrence. Si l'évènement en question consiste en un échange de message (tel est le cas dans notre scénario), d'autres informations sont enregistrées notamment l'émetteur, le récepteur, l'acte de communication, le langage, l'ontologie, le protocole, l'objet de conversation et la taille du message. L'extrait suivant illustre la trace au format XML relative à l'envoi d'un message.

```
<ExecutionTrace>
        <event number="1" type="1" time="1213429754843">
                <message>
                        <sender>INTER</sender>
                        <receiver>LOCBN</receiver>
                        <performative>INFORM</performative>
                        <language>LEAP</language>
                        <ontology>sim-ontologie</ontology>
                        <protocol />
                        <conversationId>matrice</conversationId>
                        <size>15084</size>
                </message>
        </event>
```

La trace d'exécution est exploitée ensuite par le module de mesure pour construire le graphe représentatif du SMA. L'évaluateur a la possibilité de visualiser ce graphe qui est affiché sous sa forme graphique. Une fois les mesures de performances calculées par le module de mesure, elles seront également affichées sous forme de diagrammes afin de faciliter à l'évaluateur la tâche de leur interprétation. Les résultats relatifs à ces deux dernières étapes sont détaillés dans les paragraphes suivants à travers des expérimentations sur deux applications multi-agents.

2. Application de diagnostic des pannes

Nous avons choisi de mener nos expérimentations d'abord sur une application de diagnostic pour mettre en valeur l'utilité de notre travail pour le monde industriel où les enjeux de performances sont très importants et où les notions de coût, de productivité et d'efficacité sont des objectifs primordiaux. Le paragraphe suivant est dédié à la présentation de cette application.

2.1. Présentation de l'application

L'application sur laquelle nous testons notre module d'observation est une application multi-agents destinée à la détection et à la localisation des défaillances au niveau des variables de surveillance d'un système industriel [Sad07], [Bou07]. Le principe de fonctionnement de cette application se résume comme suit : on dispose d'un modèle analytique décrivant le fonctionnement normal du système. Le diagnostic du système réel se fait en testant la cohérence des observations effectuées sur ce dernier avec le modèle analytique à travers un ensemble d'indicateurs de défaillance ou résidus. Le processus de diagnostic se déroule en deux phases : la première est la phase de détection qui consiste à calculer les valeurs de résidus et de les tester pour savoir si une panne existe. La deuxième phase est la phase de localisation, elle se fait en comparant les valeurs des résidus générés aux vecteurs binaires correspondant aux différents défauts de fonctionnement pour déterminer la source de la panne. L'application multi-agents de diagnostic se compose de plusieurs agents autonomes, chacun de ces agents est responsable d'une tâche spécifique et peut appartenir à une des classes suivantes :

- **Les agents de détection** : ils récupèrent les valeurs des observations effectuées sur le système à diagnostiquer et calculent les valeurs des différents résidus. Deux méthodes de détection sont utilisées : une méthode basée sur la logique binaire et une méthode basée sur la logique floue. Dans la première méthode, à chaque résidu est associé un agent qui se charge de calculer sa valeur et décide si cette dernière correspond à un état normal ou à un état défectueux. Alors que dans la deuxième méthode, un seul agent se charge de traiter tous les résidus et de déterminer si l'ensemble du système est dans un état normal ou dans un état de défaillance.

- **Les agents de localisation** : leur rôle consiste à récupérer les différentes valeurs des résidus produits par les agents de détection et à localiser la défaillance en utilisant différentes méthodes. Les méthodes utilisées sont la logique binaire, la logique floue, les réseaux de neurones probabilistes et le perceptron multicouche. A chaque méthode

de localisation est associé un agent qui en fait usage. Alors que les agents utilisant la première et la deuxième méthode ont besoin d'informations des agents de détection pour localiser la défaillance, les agents utilisant la troisième et la quatrième méthode sont en mesure de calculer et de traiter les résidus et ainsi de localiser les défaillances sans recours à un agent de localisation.

- **L'agent d'interfaçage** : son rôle consiste d'une part à coordonner les actions des autres agents en contrôlant l'ordre des échanges des messages, et d'une autre part à afficher à l'utilisateur les interfaces graphiques à travers lesquelles il peut interagir et être informé de l'état du système.

- **L'agent arbitre** : il permet de gérer les situations de conflit entre les différents agents de localisation. Si ces derniers aboutissent à des résultats différents, l'agent arbitre permet de trancher en se référant à l'historique de chacun d'entre eux. Pour ce faire, à chaque agent de localisation est associé un degré de crédibilité proportionnel au nombre de diagnostics corrects qu'il a émis précédemment.

L'application de diagnostic des pannes se décline en deux versions :

- Une première version qui ne prévoit rien pour la gestion des conflits au cas où les résultats de localisation des différents agents ne sont pas cohérents. Cette version ne fait donc pas intervenir l'agent d'arbitrage.
- Une seconde version qui fait intervenir un agent arbitre en plus de tous les autres agents qui assurent les tâches de détection et de localisation des pannes.

Les deux versions de cette application ont été implémentées sur la plateforme multi-agents JADE[4] qui est une plateforme standard conforme aux spécifications FIPA[5] (Foundation for Intelligent Physical Agents). Tous les agents de l'application coopèrent et communiquent par échange de messages ACL[6] (Agent Communication Language) pour assurer les différentes tâches du diagnostic.

Les figures 4.2 et 4.3 ci-après présentent des schémas décrivant les différents agents utilisés dans les deux versions de cette application et les différentes interactions qui les lient.

[4] http://jade.tilab.com/

[5] http://www.fipa.org/

[6] http://www.fipa.org/repository/aclspecs.php3

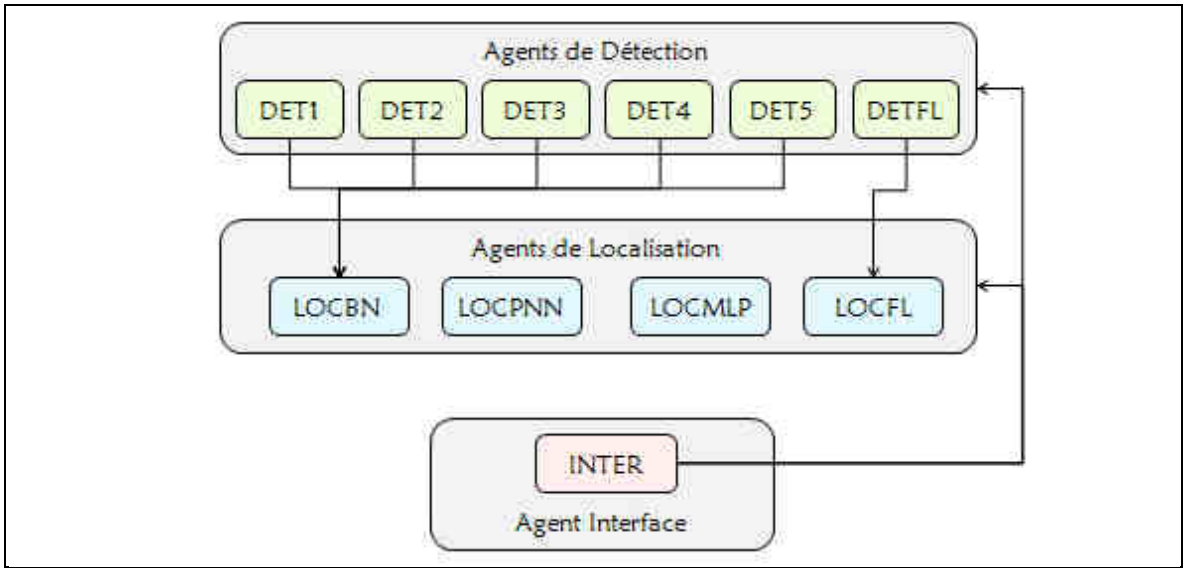

Figure 4.2 : Schéma descriptif de la première version de l'application de diagnostic des pannes

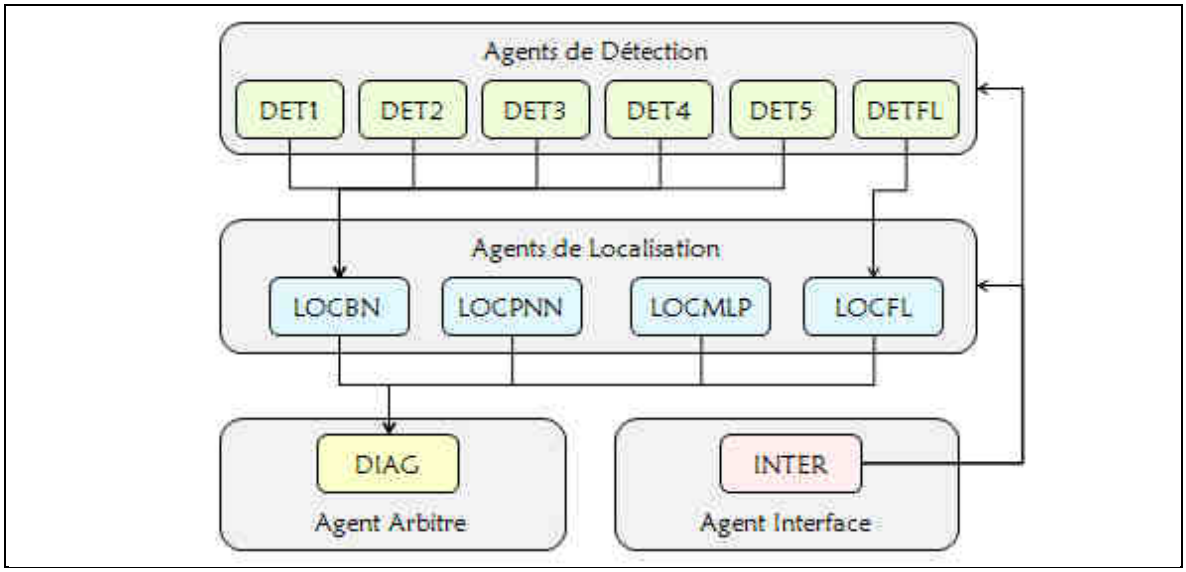

Figure 4.3 : Schéma descriptif de la seconde version de l'application de diagnostic des pannes

DET1, DET2, DET3, DET4 et DET5 sont les agents de détection qui se basent sur la logique binaire. DETFL est l'agent de détection qui se base sur la logique floue. LOCBN est l'agent de localisation qui utilise la logique binaire. LOCFL est celui qui utilise la logique floue. LOCPNN utilise la méthode de localisation basée sur les réseaux de neurones probabilistes. LOCMLP utilise la méthode de localisation basée sur le perceptron multicouche. INTER est l'agent d'interfaçage, et enfin DIAG est l'agent arbitre.

2.2. Résultats et interprétations

Dans cette partie nous présentons les résultats obtenus suite aux tests d'évaluation effectués sur les deux versions de l'application de diagnostic.

2.2.1. Graphes de communication

La figure 4.4 suivante représente le graphe de communication de la première version de l'application de diagnostic.

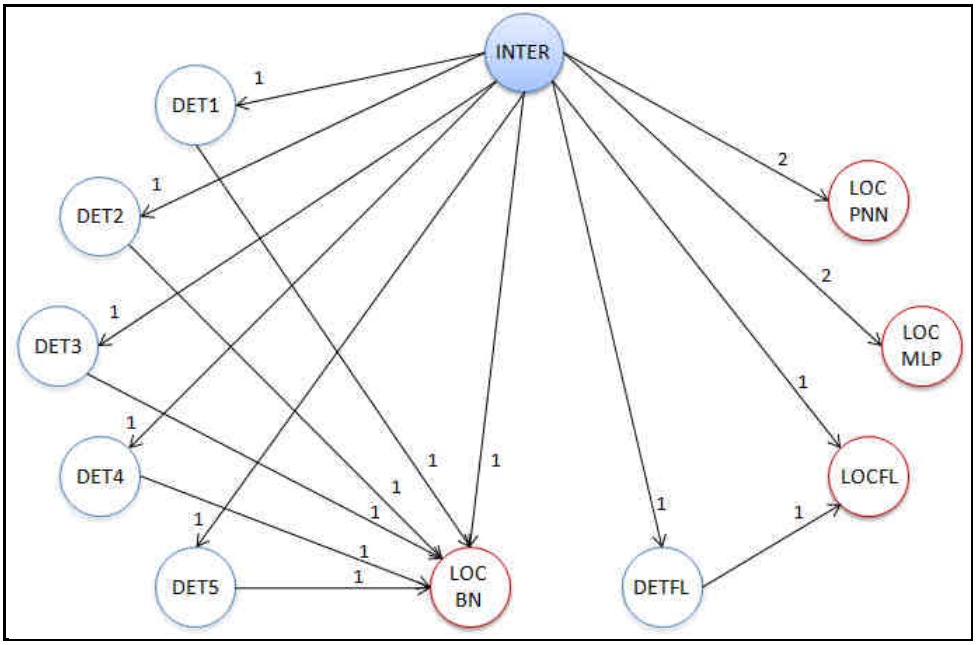

Figure 4.4 : Graphe de communication de la première version de l'application

En se basant sur ce, nous remarquons que les agents de détection DET1-5 et DETFL ne communiquent pas ensemble, ils s'en remettent aux agents de localisation LOCBN et LOCFL qui sont plus sollicités que les autres agents. Ceci est dû au fait que lorsqu'une panne est détectée les valeurs des résidus sont envoyées aux agents de localisation. Au contraire, l'agent interface INTER envoie des messages et n'en reçoit pas, ceci est dû au fait que c'est un agent organisateur, il transmet aux agents de détection une requête de diagnostic et informe les agents de localisation qu'ils devraient recevoir les résultats de la part des agents de détection.

La figure 4.5 suivante représente le graphe de communication de la deuxième version de l'application de diagnostic.

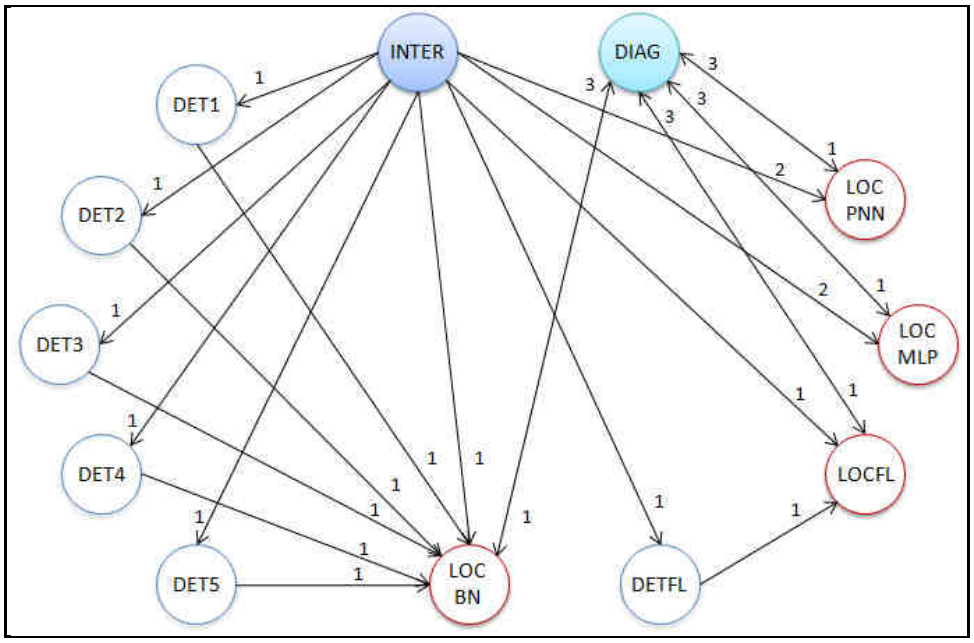

Figure 4.5 : Graphe de communication de la deuxième version de l'application

Le graphe souligne la présence d'un nouvel agent DIAG dans cette seconde version. Il s'agit d'un agent arbitre qui aide l'utilisateur dans la prise de décision en cas de conflit entre les différents agents de localisation.

2.2.2. Evaluation de la communication

Nous présentons dans ce qui suit les résultats obtenus en se basant sur l'étude des graphes de communication générés pour les deux versions de l'application de diagnostic.

Le tableau 4.1 suivant regroupe les différentes mesures obtenues pour la première version de l'application.

Mesure	Valeur
Indice β	1.4545
Indice γ	0.1322
Indice θ	1.6363
Nombre de composantes connexes	1
Nombre de points d'articulation	1
Nombre de messages échangés	18
Nombre d'agents communicants	11
Pourcentage d'agents communicants	100%

Tableau 4.1 : Valeurs des mesures pour la première version de l'application

En se basant sur ces résultats, nous constatons que le réseau de communication n'est pas complexe, l'indice β est faible et γ est plus proche de 0 que de 1. Ainsi le SMA est caractérisé par un faible degré de communication et une faible connectivité. Tout le SMA consiste en un seul réseau d'accointance dans lequel tous les agents participent à la communication.

A travers les figures 4.6, 4.7 et 4.8 ci dessous, nous remarquons que la charge de communication n'est pas partagée équitablement et que les agents LOCBN et INTER sont impliqués dans la communication plus que les autres, ce qui peut révéler une certaine centralisation du contrôle. Une alternative possible au problème de charge serait de répliquer les agents concernés et d'associer à chacun un sous-ensemble d'agents de détection dans le but de réduire les risques de disfonctionnement de l'application.

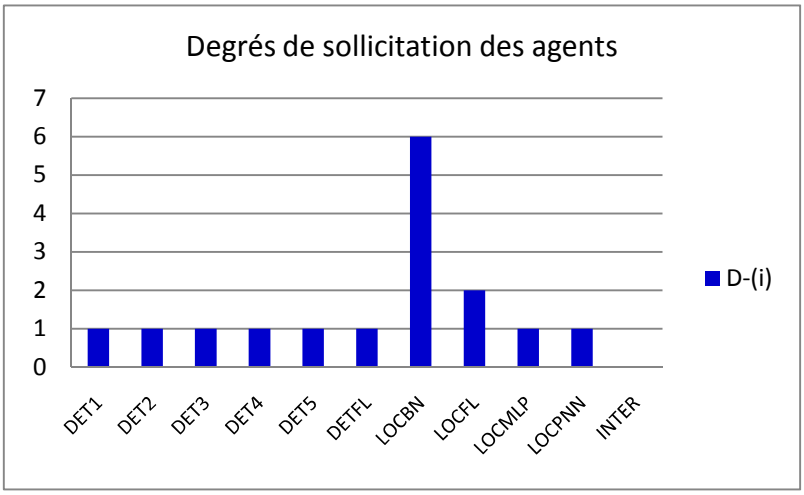

Figure 4.6 : Degrés de sollicitation des agents

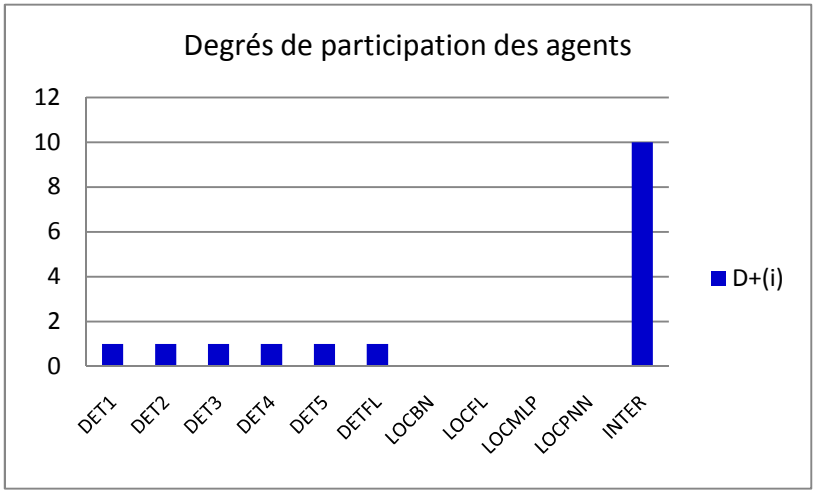

Figure 4.7 : Degrés de participation des agents

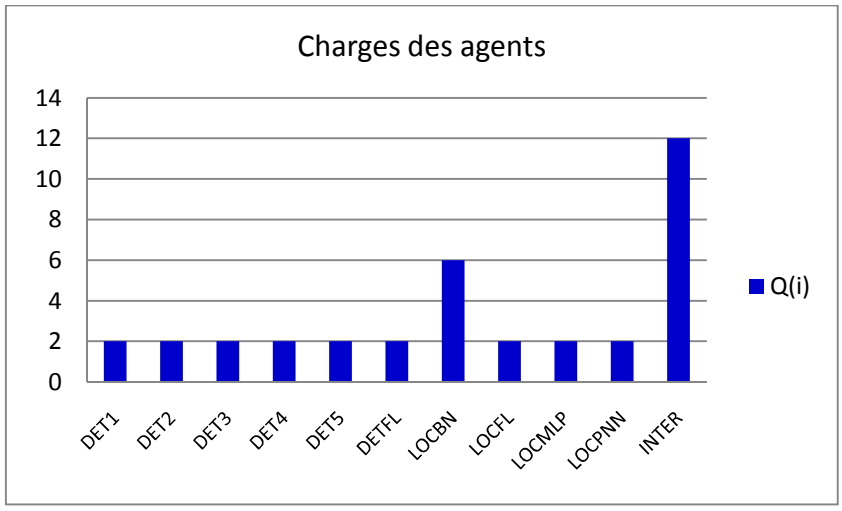

Figure 4.8 : Charges des agents

Comme illustré par la figure 4.9, la typologie des messages est pauvre, les agents utilisent un seul type de messages qui est représenté par la performative INFORM. Cela semble naturel au vu des besoins de l'application. En effet, la communication entre les différents agents consiste en un échange d'information, il n'y a pas de conversations complexes et sophistiquées.

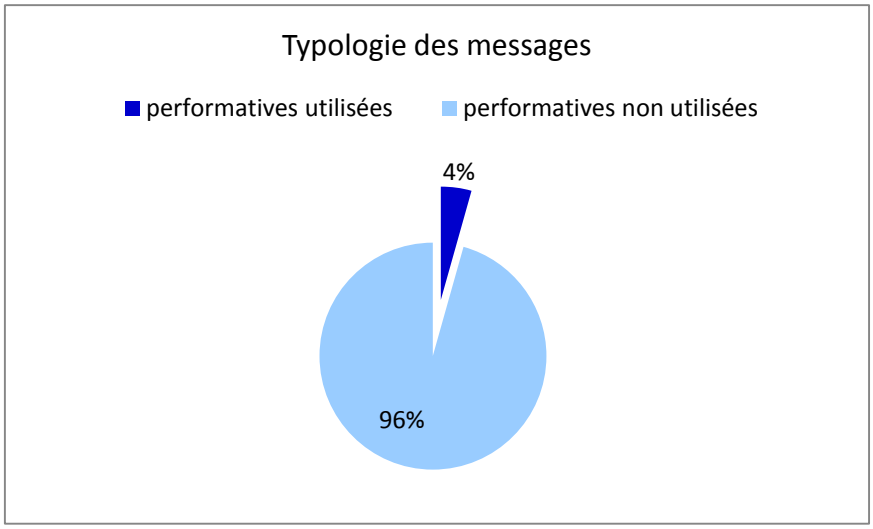

Figure 4.9 : Typologie des messages

D'après la figure 4.10 la taille des messages est sensiblement la même. Tous les contenus des messages sont de complexité moyenne du fait que les agents s'envoient des matrices encodées respectant une ontologie définie par le concepteur de l'application.

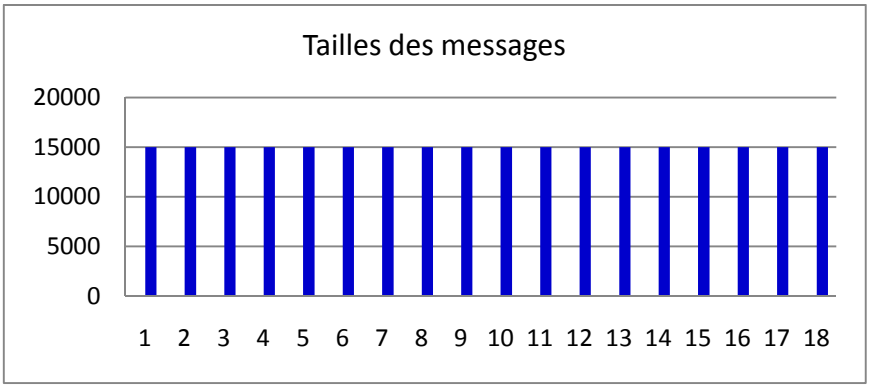

Figure 4.10 : Tailles des messages

Le tableau 4.2 suivant regroupe les différentes mesures obtenues pour la seconde version de l'application de diagnostic.

Mesure	Valeur
Indice β	2
Indice γ	0.1667
Indice θ	2.3334
Nombre de composantes connexes	1
Nombre de points d'articulation	0
Nombre de messages échangés	34
Nombre d'agents communicants	12
Pourcentage d'agents communicants	100%

Tableau 4.2 : Valeurs des mesures pour la deuxième version de l'application

En se basant sur les valeurs de ces mesures, nous remarquons que le réseau de communication n'est pas complexe dans ce cas aussi. L'indice γ est plus proche de 0 que de 1 donc le SMA est caractérisé par un faible degré de communication. De plus, tout le SMA consiste en un seul réseau d'accointance dans lequel tous les agents participent à la communication.

Nous constatons, néanmoins, que l'introduction du nouvel agent DIAG a augmenté le niveau de communication par rapport à la première version de l'application. Cette augmentation est soulignée par la modification des valeurs de certains indices. En effet, la complexité du réseau de communication représentée par l'indice β, est passée de 1.45 à 2. Cependant, elle reste toujours une valeur faible.

La même chose est constatée pour le degré de communication représenté par l'indice γ. Bien qu'il soit passé de 0.13 à 0.16 il demeure toujours faible. L'indice θ, quant à lui, est passé de 1.63 à 2.33 ce qui traduit une augmentation de l'interactivité du SMA, une dynamique qui a été renforcée par la présence du mécanisme d'arbitrage à travers l'agent DIAG.

En se basant sur les figures 4.11, 4.12, 4.13, nous remarquons que la charge de communication n'est pas équitablement répartie et que cette fois-ci c'est les agents INTER et DIAG qui sont impliqués dans la communication plus que les autres du fait qu'il s'agit d'agents chargés de la coordination.

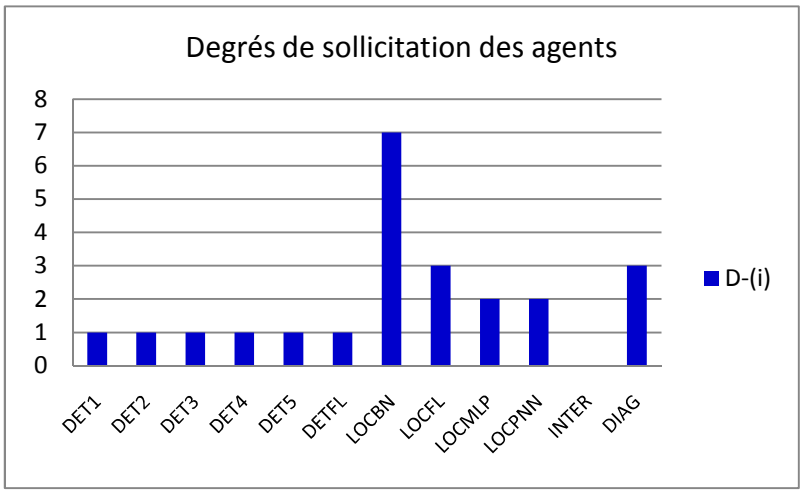

Figure 4.11 : Degrés de sollicitation des agents

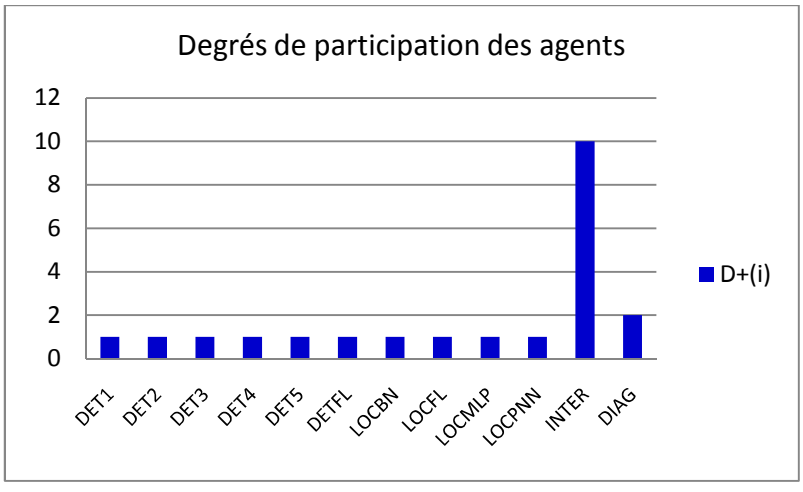

Figure 4.12 : Degrés de participation des agents

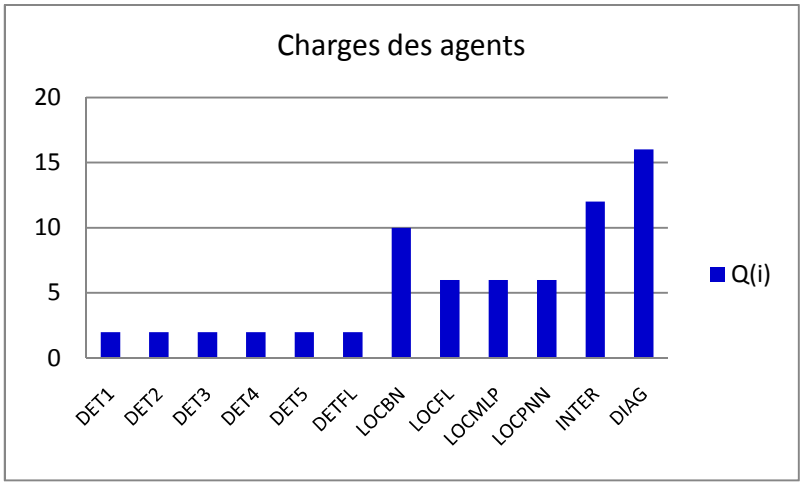

Figure 4.13 : Charges des agents

Comme illustré par la figure 4.14, la typologie des messages reste pauvre, les agents utilisent deux types de messages représentés par les performatives INFORM et PROPOSE. Nous remarquons donc que la typologie de la communication est plus riche dans cette seconde version, cela est dû à l'utilisation de la performative supplémentaire PROPOSE dans le processus d'arbitrage.

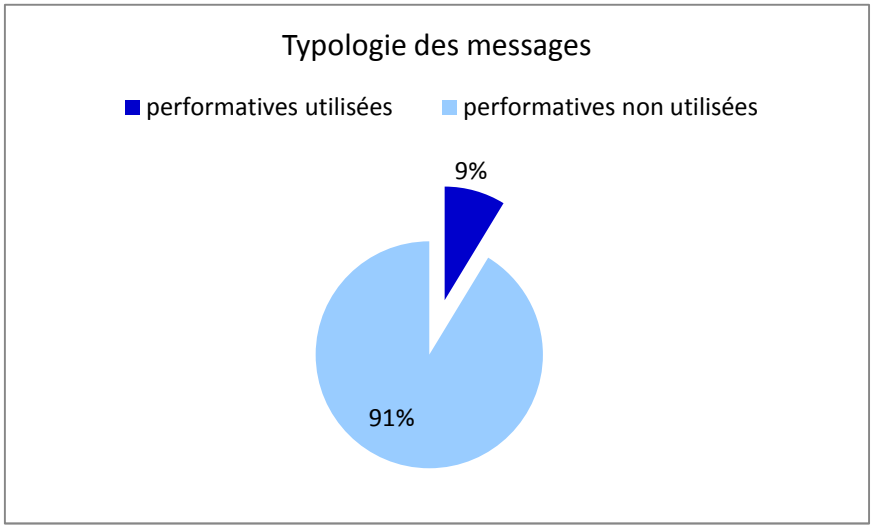

Figure 4.14 : Typologie des messages

La figure 4.15 ci-après montre que les messages correspondant à la communication induite par l'ajout de l'agent arbitre sont de taille sensiblement inférieure à celle des messages échangés entre le reste des agents du SMA. En effet, ces messages permettent de transmettre des informations concises et utiles pour la résolution des conflits et ne contiennent pas de données caractéristiques liées aux processus de détection des pannes.

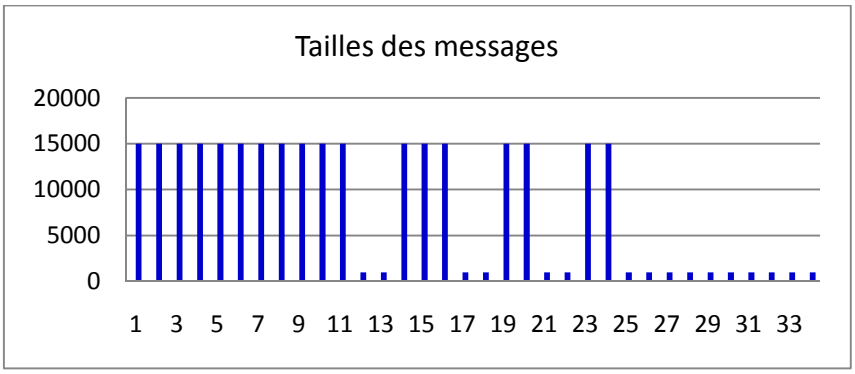

Figure 4.15 : Tailles des messages

Nous remarquons qu'il y a plus de communication dans la seconde application. Le nombre de messages échangés a augmenté. Les impacts sur les performances de l'application semblent être importants. Toutefois, si on examine les tailles des messages et leur complexité on remarque que les messages supplémentaires échangés par l'agent DIAG sont de tailles réduites et non complexes car il s'agit de simples messages textes. Ainsi, nous pouvons conclure que l'impact de l'ajout d'un agent arbitre avec tout ce que cela engendre au niveau de la communication et des mesures de performances qui s'y afférent reste négligeable.

2.2.3. Evaluation de l'organisation

La figure 4.16 suivante exprime les degrés *Ki* des agents pour les deux versions de l'application.

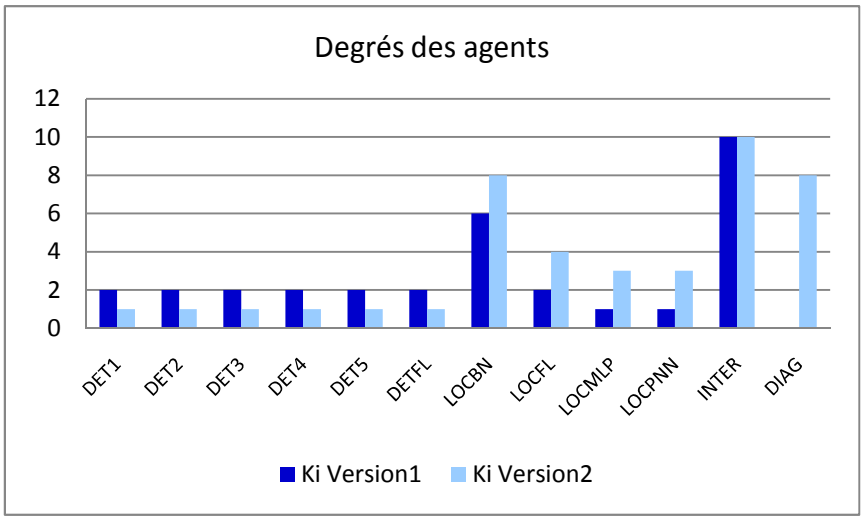

Figure 4.16 : Degrés des agents

Nous distinguons deux catégories d'agents :

- Des agents dont le degré a diminué ou n'a pas changé d'une version à l'autre tels que les agents de détection et l'agent d'interfaçage ;
- Des agents dont le degré a augmenté suite à l'ajout d'un agent arbitre tels que les agents de localisation.

Ceci est dû au fait que les agents de localisation sont ceux qui émettent les résultats de diagnostic et qui risquent le plus de se retrouver dans des situations conflictuelles d'où leur recours à l'agent d'arbitrage.

La figure 4.17 ci-dessous illustre l'aspect des distributions des degrés correspondants aux différentes versions de l'application de diagnostic.

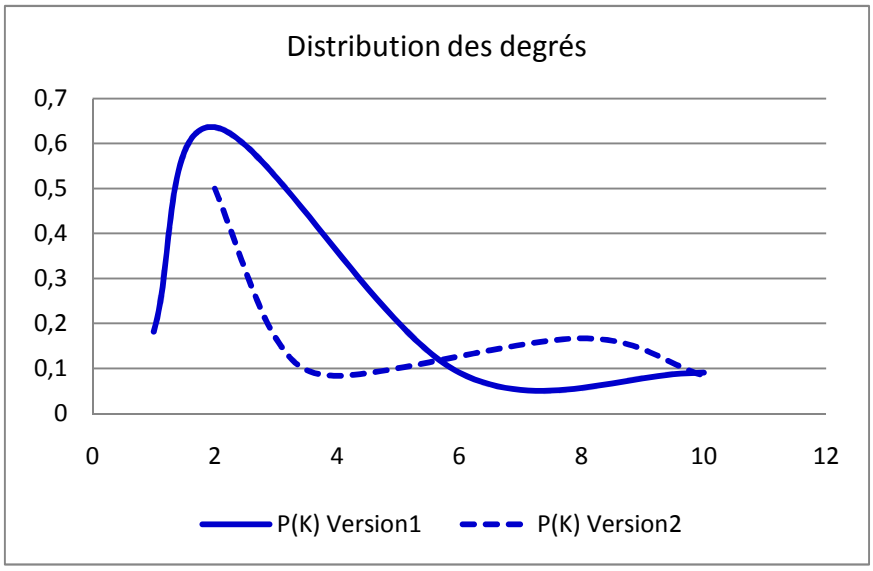

Figure 4.17 : Distribution des degrés

Les courbes obtenues correspondent à des graphes hétérogènes, c'est-à-dire qu'il n'y a pas de valeur moyenne significative autour de laquelle les degrés sont concentrés. Dans les deux cas, il y a une majorité de nœuds ayant des degrés faibles et un petit nombre de nœuds avec un degré important qui excède la moyenne. Nous remarquons également que l'écart entre le nombre d'agents ayant un degré faible et celui d'agents ayant un degré important est beaucoup plus accentué dans la première version de l'application (sans arbitrage).

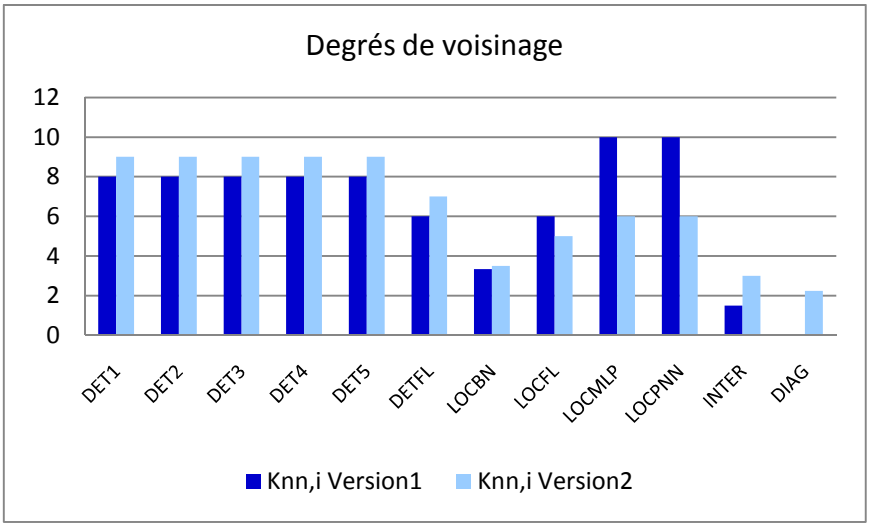

Figure 4.18 : Degrés de voisinage des agents

C'est également ce que nous constatons à travers la figure 4.19 suivante qui illustre les corrélation-degrés pour les deux versions de diagnostic.

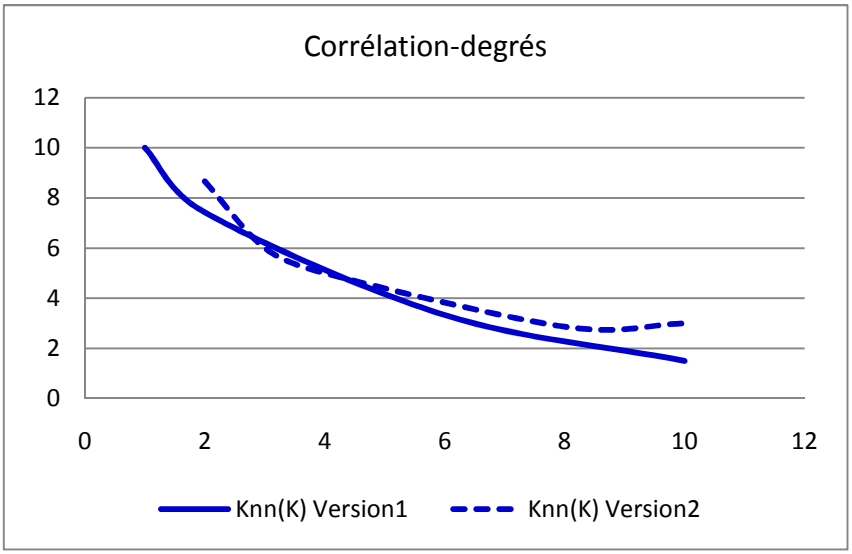

Figure 4.19 : Corrélation-degrés (mesure de l'assortativité)

Le corrélation-degré exprime l'assortativité du graphe. Les deux architectures sont disassortatives. La disassortativité est une caractéristique des systèmes hiérarchiques dans lesquels les agents de degrés importants tendent à s'associer aux agents de degré moins importants. Nous remarquons que l'assortativité dans le cas sans arbitrage est légèrement différente par rapport au cas avec arbitrage. Dans ce dernier, la disassortativité est accentuée par l'ajout de l'agent arbitre dont le degré est important par rapport aux autres agents.

La figure 4.20 suivante illustre les degrés de centralité des différents agents pour les deux versions de diagnostic.

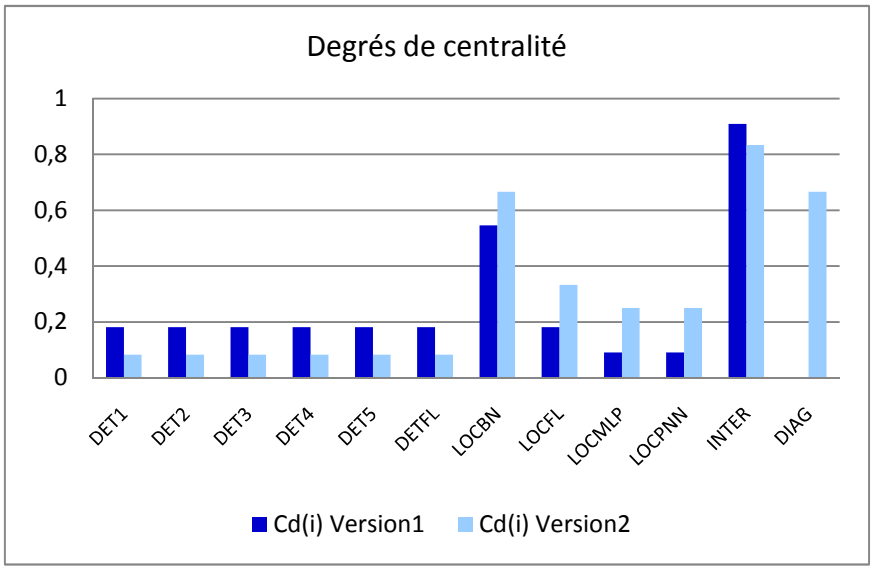

Figure 4.20 : Degrés de centralité des agents

Nous remarquons à travers ces résultats que les degrés de centralité des agents de détection ont diminué en introduisant l'agent arbitre et les degrés des agents de localisation ont au contraire augmenté. Ceci s'explique par le fait que c'est les agents de localisation qui sont en contact avec l'agent arbitre ce qui accentue la centralisation au sein de cette accointance du SMA.

3. Application de gestion de la production

3.1. Présentation de l'application

Cette application multi-agents est un modèle simulé d'un cas d'étude proposé dans [Fra07]. Comme le montre la figure 4.21 ci-après, il s'agit d'un réseau de chaînes logistiques dans le domaine de la fabrication des meubles. Deux familles de produits sont fabriquées : les tables et les étagères. Dans une première étape les troncs de bois, délivrés par les fournisseurs de bois sont transformés dans les scieries en divers composants tels que les plateaux, les planchettes et les pieds. En second lieu, les tables et les étagères sont assemblées dans l'entreprise d'assemblage. Les étagères sont directement livrées aux clients finaux et les tables sont transportées à l'entreprise de peinture qui les peint dans une dernière étape et les délivre aux clients finaux.

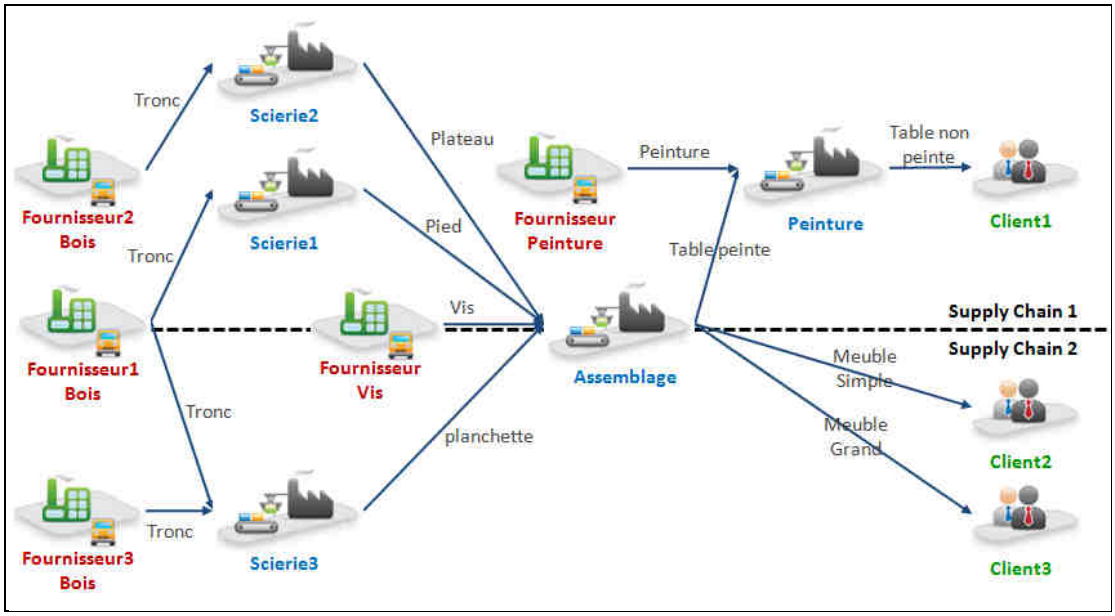

Figure 4.21 : Cas d'étude, réseau de chaînes logistiques

Chaque entreprise de la chaîne logistique se compose de quatre services : la distribution, la production, l'approvisionnement et la prise de décision qui sont représentés dans le modèle proposé par les agents suivants [Ben12a] :

- **Distribution :** cet agent se charge de la réception et de la gestion des commandes des clients. Il se charge également de la gestion des stocks de produits et des livraisons des commandes.

- **Production :** cet agent gère les flux de produits et assure la production effective, il transforme les produits semi-finis ou la matière première reçue en produits en se basant sur des règles de nomenclature spécifiés.

- **Approvisionnement :** cet agent se charge de contacter les fournisseurs et de passer les commandes chaque fois qu'il ya besoin de matière première ou de produits semi-finis pour assurer le processus de production.

- **Décision :** cet agent synchronise les actions des autres agents de l'entreprise. Il assure aussi la planification de la production et des ressources nécessaires. La distribution des différents agents de décision à travers la chaîne logistique et les fonctions qui leurs sont associées définissent l'architecture de pilotage. Dans le cas du pilotage distribué, chaque centre de décision est responsable de la prise de décision dans son entreprise,

alors que dans le cas des pilotages centralisé et mixte, les centre de décision de l'entreprise s'en remettent à un décideur central d'un plus haut niveau.

En plus de ces agents, la chaîne logistique contient les clients finaux et les fournisseurs de matières premières qui sont représentés par les agents suivants :

- **Client :** cet agent représente un client final désireux de se procurer un produit fini bien déterminé.

- **Fournisseur :** cet agent représente un fournisseur de matière première.

La gestion des plans de production, d'approvisionnement et de distribution nécessite la mise en place d'un système de prise de décision tout en définissant l'organisation des différents centres de décision à travers le réseau des chaînes logistiques. Le système décisionnel peut être organisé selon trois architectures de pilotage : le pilotage distribué, le pilotage centralisé et le pilotage mixte.

Dans le pilotage distribué, il y a un unique niveau décisionnel. Chaque entreprise de la chaîne logistique est totalement indépendante et gère localement ses propres ressources et productions à travers son centre de décision.

Pour remédier au manque de visibilité globale de l'information engendré par le pilotage distribué, une alternative consiste à rendre cette information accessible à tous les acteurs de la chaîne et d'en centraliser le traitement. En effet, dans le pilotage centralisé, il y a un niveau décisionnel supplémentaire. Ce niveau comporte un seul décideur central qui se charge de la planification de la production et des ressources de tout le réseau des chaînes logistiques. Les autres centres de décision de niveau inférieur sont réduits à de simples agents relayant l'information au décideur central.

Dans l'objectif de trouver un compromis entre l'autonomie locale engendrée par le pilotage distribué et l'optimisation globale apportée par le pilotage centralisé, une troisième stratégie de pilotage est proposée dans [Fra07]. IL s'agit du pilotage mixte qui permet d'avoir plusieurs décideurs centraux de plus haut niveaux au lieu d'un seul comme dans le cas du pilotage centralisé. En effet, à chaque chaîne logistique du réseau est associé un décideur qui centralise l'information et le traitement relatifs à la gestion de la production de toute la chaîne. L'échange d'information entre les décideurs centraux et les centres de décision de niveau plus bas sont les mêmes que dans le cas du pilotage centralisé.

3.2. Résultats et interprétations

Les expérimentations réalisées sur les trois version de l'application décrites ci-dessus ont permis de tirer les résultats présentés dans les paragraphes suivants.

3.2.1. Graphes de communication

La figure 4.22 ci-après montre le graphe de communication correspondant à l'architecture de pilotage distribué. Chacun des centres de décision S1.dec, S2.dec, S3.dec, ASS.dec et PNT.dec se charge localement de la gestion de la production et des ressources de l'entreprise à laquelle il appartient.

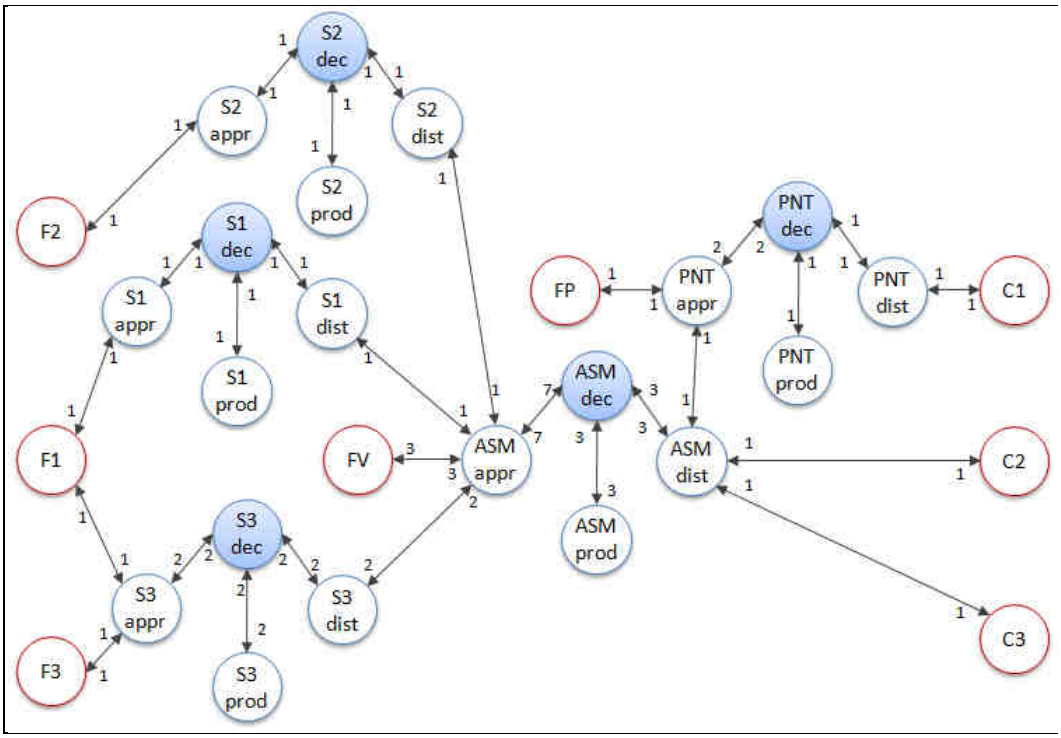

Figure 4.22 : Graphe représentatif de l'architecture de pilotage distribué

La figure 4.23 montre le graphe de communication correspondant à l'architecture de pilotage centralisé. Dans ce cas, il y a un décideur central de plus haut niveau qui réalise la gestion de la production de tout le réseau des chaînes logistiques.

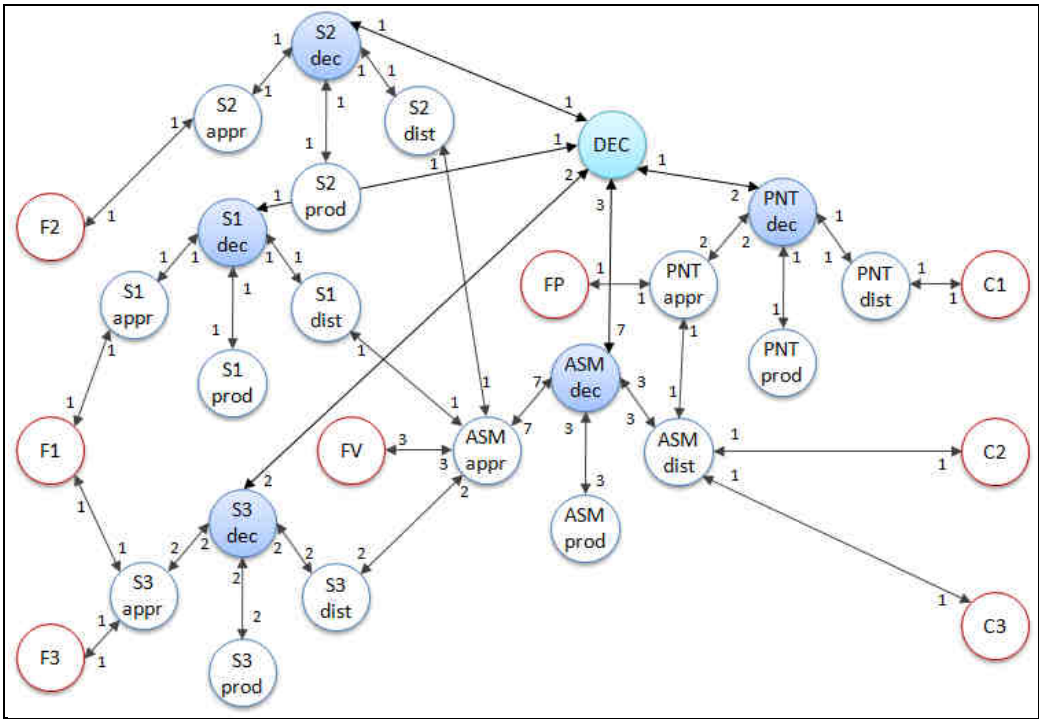

Figure 4.23 : Graphe représentatif de l'architecture de pilotage centralisé

La figure 4.24 montre le graphe de communication correspondant à l'architecture de pilotage mixte. Dans ce cas, il y a un décideur en plus pour chaque chaîne logistique.

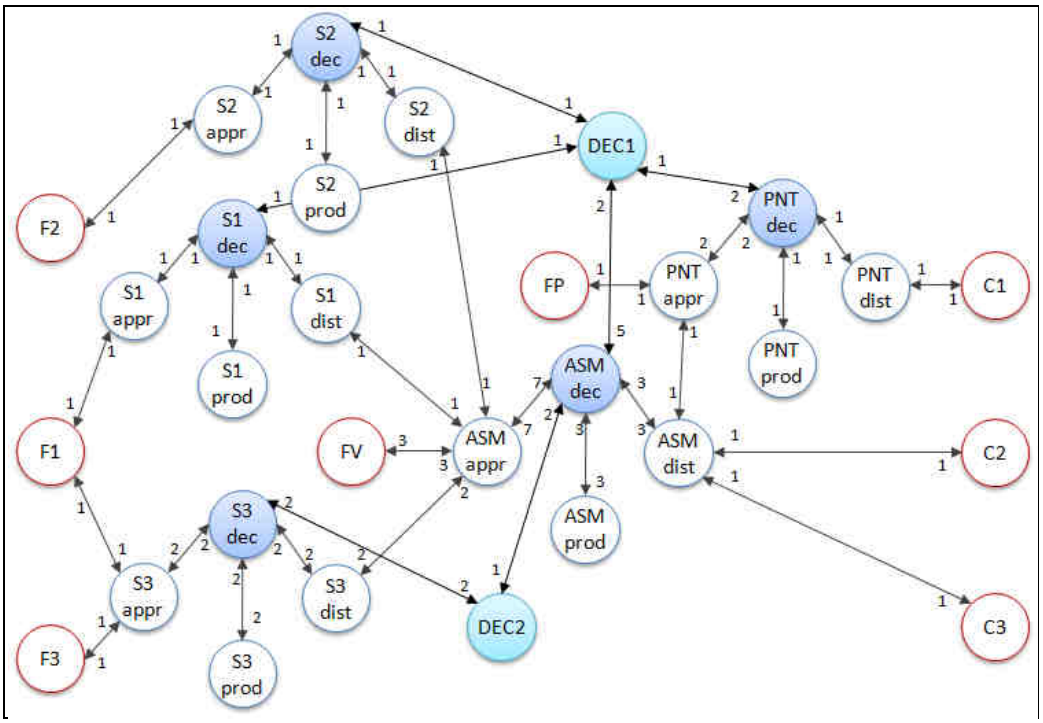

Figure 4.24 : Graphe représentatif de l'architecture de pilotage mixte

3.2.2. Evaluation de la communication

Nous présentons dans ce qui suit les résultats obtenus en se basant sur l'étude des graphes de communication générés pour les trois versions de l'application de pilotage.

Mesure	Cas distribué	Cas centralisé	Cas mixte
Indice β	1	1.1379	1.1333
Indice γ	0.0740	0.0812	0.0781
Indice θ	3.2857	3.8965	3.7666
Nombre de composantes connexes	1	1	1
Nombre de points d'articulation	15	15	15
Nombre de messages échangés	92	113	113
Nombre d'agents communicants	28	29	30
Pourcentage d'agents communicants	100%	100%	100%

Tableau 4.3 : Valeurs des mesures des trois versions de l'application

Les valeurs présentées dans le tableau 4.3 ci-dessus mettent en évidence les différences entre les diverses architectures de pilotage en termes de communication. L'indice β, traduisant la complexité du réseau de communication, est faible dans tous les cas, néanmoins nous remarquons qu'il s'accroît avec la centralisation. Il en va de même pour l'indice γ qui traduit l'interactivité du SMA : il est très faible pour les trois architectures. La charge du réseau de communication, caractérisée grâce à l'indice θ, s'accroît également avec la centralisation. Ainsi, pour ces trois indices, les valeurs correspondantes à l'architecture mixte se placent entre celles des architectures distribuée et centralisée.

Ces résultats révèlent également la présence d'un nombre considérable de points d'articulation dans les trois architectures. La nature même des chaînes logistiques souvent linéaire est à la base de ces résultats. Les points d'articulation détectés sont soit des décideurs, soit des services d'approvisionnement soit des services de distribution. Des disfonctionnements éventuels à ces niveaux là seraient fatals à toute la chaîne si des mesures alternatives en cas de problèmes ne sont pas prévues. Nous pensons dans ce cadre au mécanisme de réplication qui peut être une solution appropriée.

A travers les figures 4.25, 4.26 et 4.27 ci dessous, nous remarquons que la charge de communication n'est pas partagée équitablement et que les agents décideurs sont impliqués dans la communication plus que les autres, ce qui rend le contrôle des agents plus centralisé.

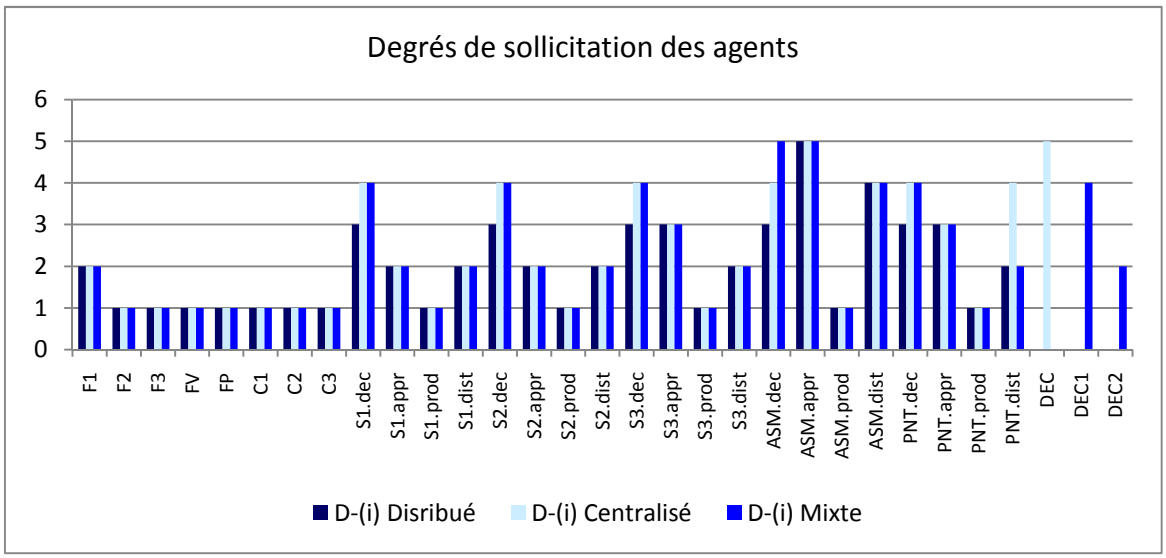

Figure 4.25 : Degrés de sollicitation des agents

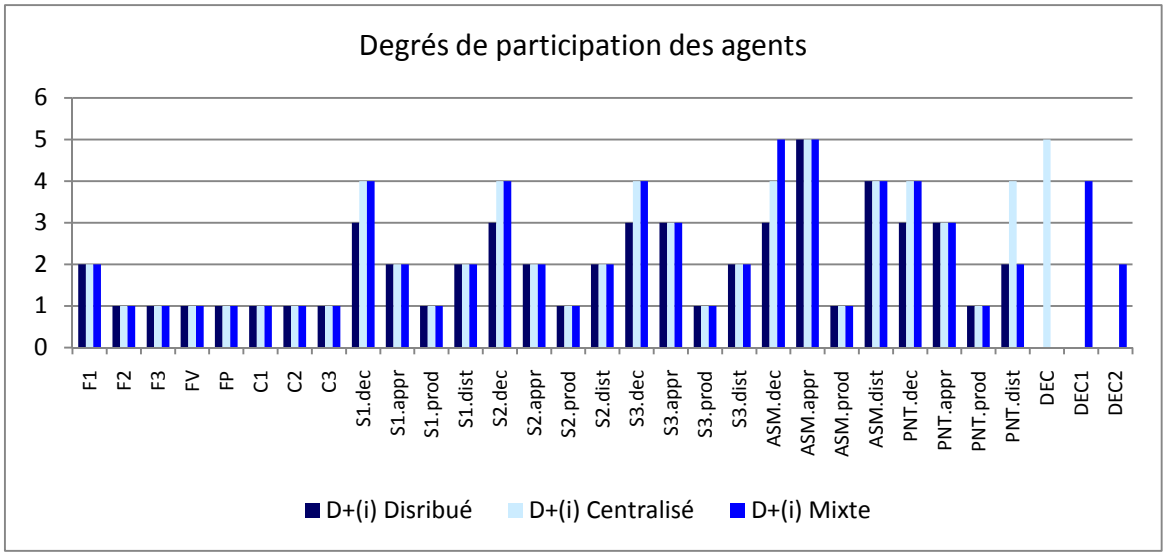

Figure 4.26 : Degrés de participation des agents

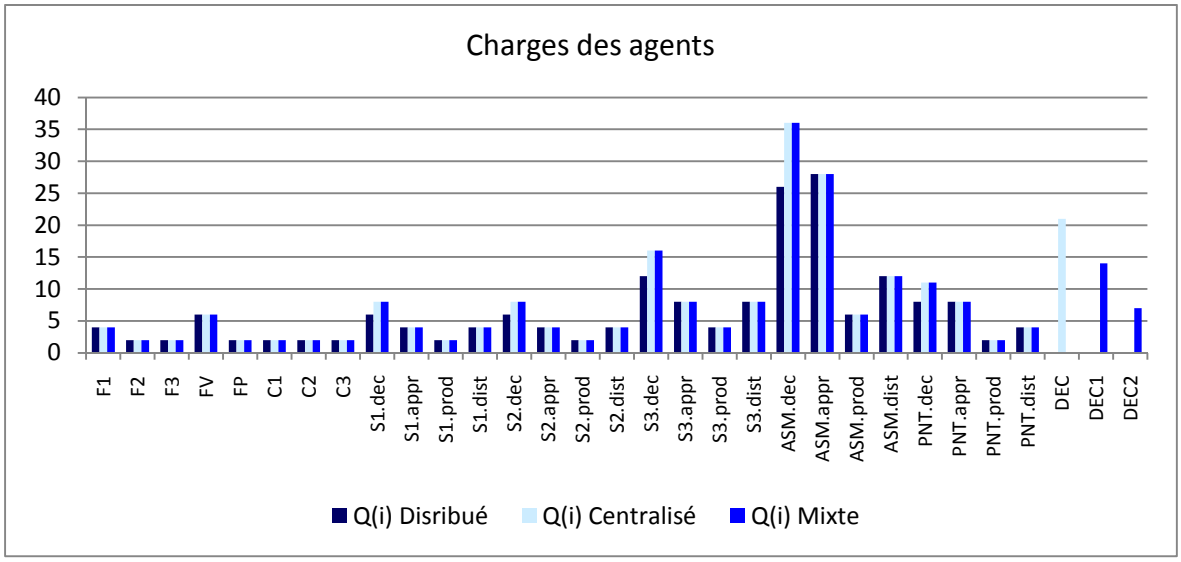

Figure 4.27 : Charges des agents

La figure 4.28 illustre la typologie des messages qui est la même pour les trois versions de l'application de pilotage et qui reste relativement pauvre car les agents utilisent quatre types de messages uniquement : REQUEST, REPLY, INFORM et ORDER. Cela semble naturel au vu des besoins de l'application. En effet, la communication entre les différents agents consiste soit en un échange de messages de types requêtes/réponses entre émetteurs et récepteurs de commandes ou bien en un échange d'information entre différents services d'une même entreprise ou alors en des ordres émis par le décideur de chaque entreprise.

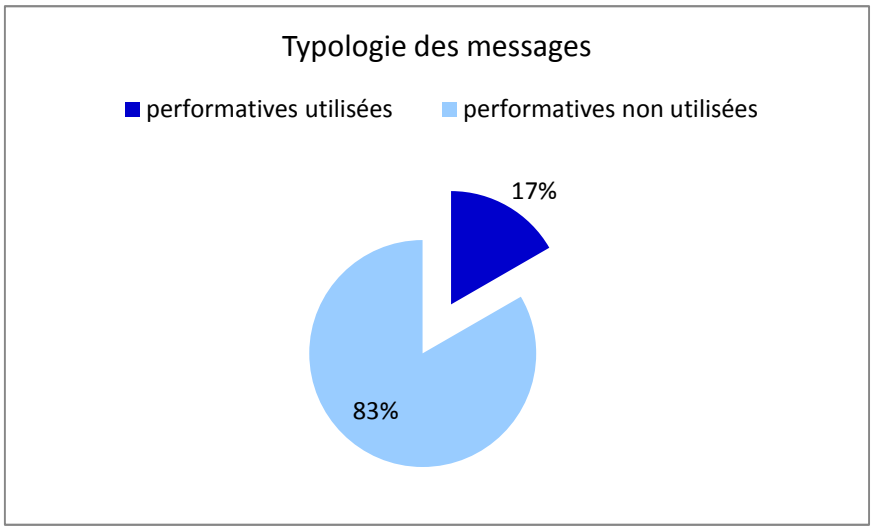

Figure 4.28 : Typologie des messages

Les figures 4.29 et 4.30 montrent respectivement le nombre et les tailles des messages pour chacune des trois versions de l'application.

D'après la figure 4.29 le nombre de messages pour les versions centralisée et mixte se rapprochent sensiblement et sont supérieurs au nombre de messages dans la version distribuée, ceci s'explique par le fait que la centralisation induit de la communication supplémentaire qui se répercute sur le nombre de messages. La moyenne des tailles des messages illustrée par la figure 4.30 reste quant à elle sensiblement la même.

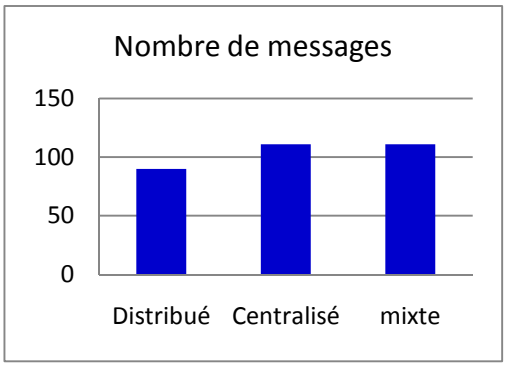

Figure 4.28 : Nombre de messages

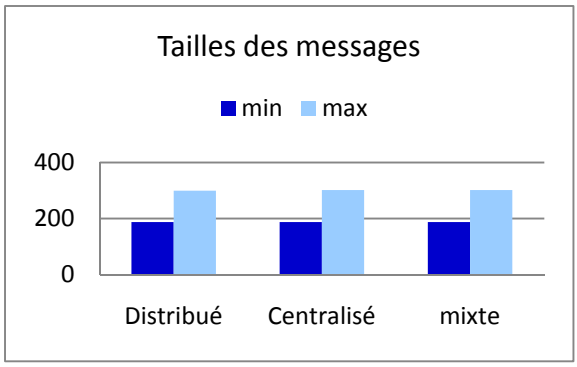

Figure 4.29 : Tailles des messages

3.2.2. Evaluation de l'organisation

La figure 4.31 suivante exprime les degrés *Ki* des agents pour les trois versions de l'application.

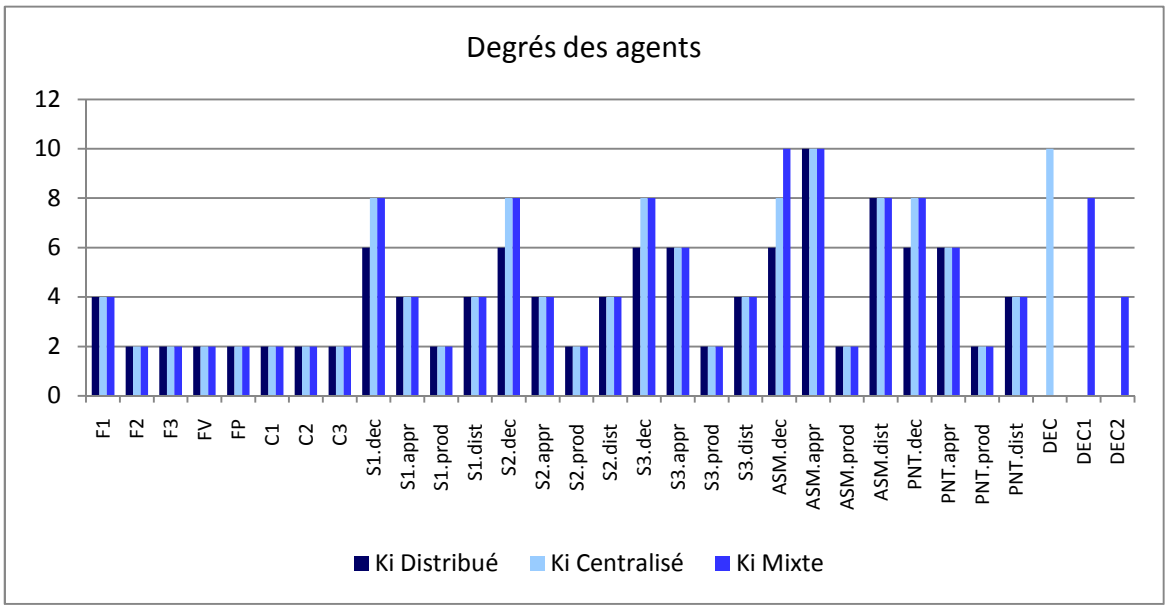

Figure 4.31 : Degrés des agents

Nous distinguons deux catégories d'agents :

- Des agents dont le degré n'a pas changé d'une architecture à l'autre tels que les clients, les fournisseurs et les services opérants dans les entreprises (production, approvisionnement et distribution) ;

- Des agents dont le degré a augmenté avec la centralisation, particulièrement dans l'architecture de pilotage mixte. L'augmentation des degrés concerne uniquement les centres de décision puisque la centralisation ne touche que le sous-système décisionnel.

De plus, nous remarquons que les agents de degrés importants sont soit des centres de décision, ou des agents de distribution ou d'approvisionnement en relation avec plusieurs entreprises.

La figure 4.32 ci-après illustre l'aspect des distributions des degrés correspondants aux différentes architectures de pilotage.

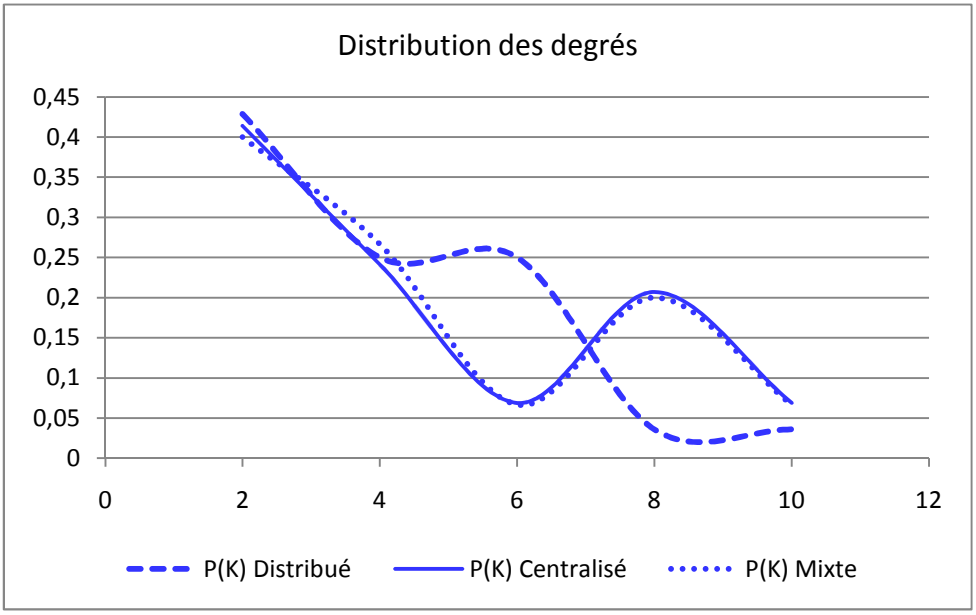

Figure 4.32 : Mesure de la distribution des degrés

Les courbes obtenues correspondent toutes à des graphes hétérogènes, c'est-à-dire qu'il n'y a pas de valeur moyenne significative autour de laquelle les degrés sont concentrés. Dans les trois cas, il y a une majorité de nœuds ayant des degrés faibles et un petit nombre de nœuds avec un degré important qui excède la moyenne. Cependant, ce que nous

pouvons constater c'est que les architectures centralisée et mixte présentent à ce niveau des caractéristiques très similaires qui se démarquent sensiblement du cas distribué.

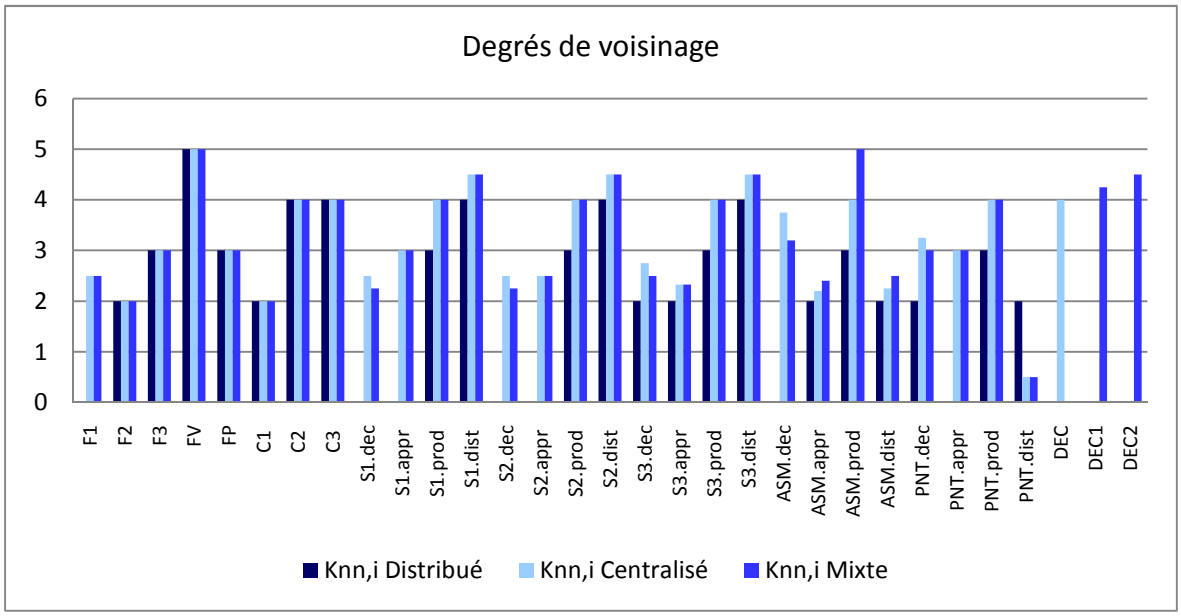

Figure 4.33 : Degrés de voisinage des agents

C'est également ce que nous constatons à travers la figure 4.34 suivante qui illustre les corrélation-degrés pour les trois architectures de pilotages.

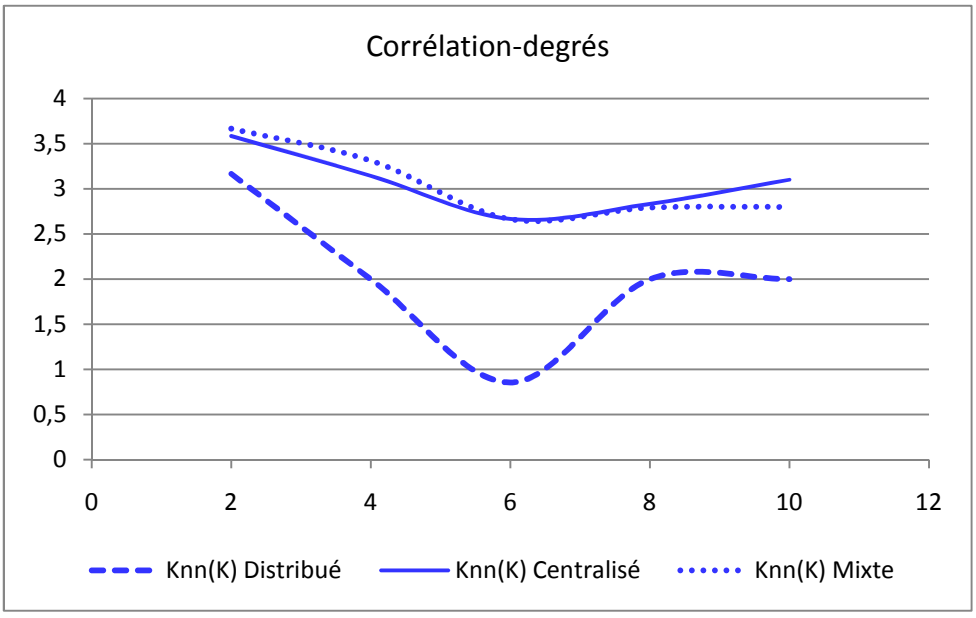

Figure 4.34 : Corrélation-degrés (mesure de l'assortativité)

Le corrélation-degré exprime l'assortativité du graphe. Les trois architectures sont disassortatives. La disassortativité est une caractéristique des systèmes hiérarchiques dans lesquels les agents de degrés importants tendent à s'associer aux agents de degrés moins importants. Ceci peut être expliqué par l'existence d'agents dominants. Ici aussi nous remarquons que l'assortativité dans le cas distribué est différente par rapport aux cas centralisé et mixte. Dans ces deux dernières, la disassortativité est accentuée par l'ajout des agents d'un niveau décisionnel supplémentaire. En effet, la relation de dominance est renforcée par la centralisation du contrôle.

La figure 4.35 suivante illustre les degrés de centralité des différents agents pour les trois architectures de pilotage.

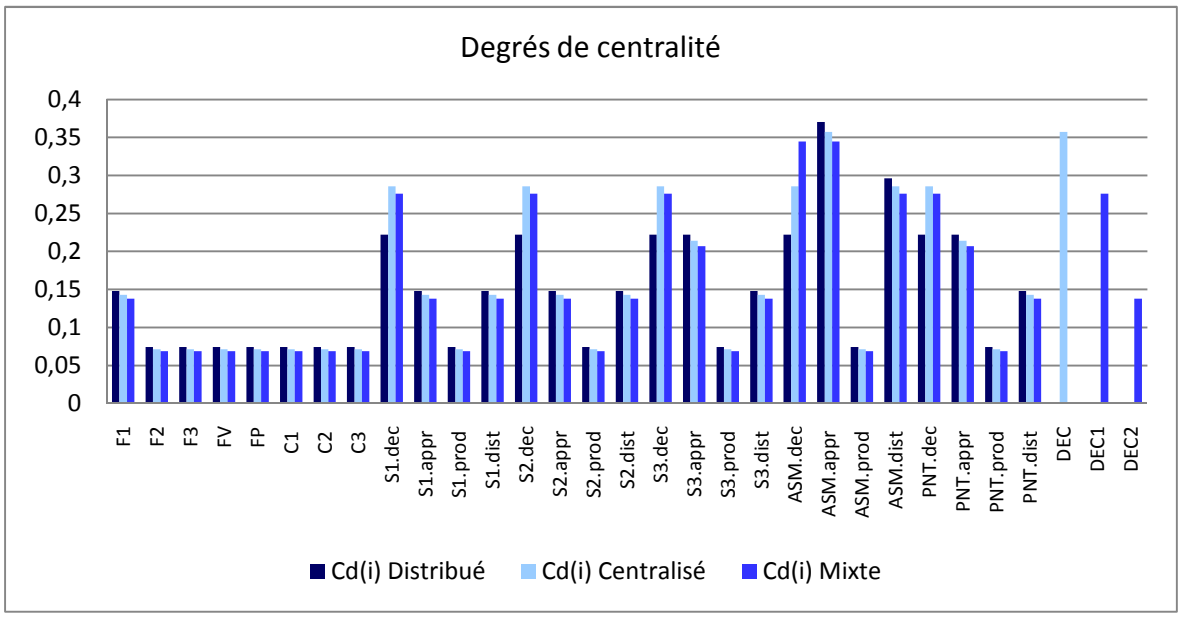

Figure 4.35 : Degrés de centralité des agents

Nous remarquons que, pour tous les agents excepté les centres de décision, la centralité diminue en passant de l'architecture distribuée aux architectures centralisée ou mixte. Au contraire, la centralité des centres de décision augmente. Ceci est dû à l'ajout d'agents dans le système qui sont liés aux centres de décision existants et pas aux autres agents.

Le tableau 4.3 ci-après présente les valeurs des degrés de hiérarchie pour chacune des architectures de pilotage simulée.

Architecture de pilotage	OrdV	MaxV	Degré de hiérarchie
Distribué	15	28	0,535
Centralisé	20	33	0,606
Mixte	21	34	0,617

Tableau 4.4 : Degrés de hiérarchie

Le degré de hiérarchie varie de 0 à 1. Dans les trois cas sa valeur est supérieure à la valeur moyenne ce qui indique que les trois applications multi-agents étudiées sont caractérisées par une tendance hiérarchique qui est d'autant plus marquée quand il s'agit des pilotages centralisé et mixte. Ceci est aussi dû à l'ajout d'agents décideurs qui ont de l'autorité sur les autres agents.

4. Bilan du travail réalisé

Au terme de cette thèse nous nous proposons d'effectuer un bilan du travail réalisé en mettant l'accent sur ses apports et en expliquant ses limites.

4.1. Apports

A travers les expérimentations que nous avons effectuées, nous avons pu valider par la pratique notre approche d'évaluation des SMA. Cette approche présente des apports intéressants à plusieurs niveaux.

Tout d'abord, notre solution a permis de répondre au besoin principal d'évaluer un SMA à son niveau applicatif en s'affranchissant de l'environnement de conception et de mise en œuvre et en tenant compte des spécificités des SMA.

L'approche d'évaluation proposée présente les avantages suivants :

- Indépendance de la méthodologie de conception et de la plate-forme d'exécution multi-agents : les solutions d'observation, de modélisation et de mesure ne sont pas restreintes à un environnement particulier et peuvent donc être adoptées pour une large classe de SMA.
- Finesse de la granularité de l'observation : l'approche proposée permet d'observer des informations à des niveaux de granularité différents qui dépendent des caractéristiques à évaluer. Ces informations peuvent aller jusqu'à des niveaux très fins telles que les variables de l'état interne des agents.

- Précision de l'observation : notre solution permet de n'observer que ce qui est nécessaire à l'évaluation d'une caractéristique bien déterminée. Il n'y a pas d'informations superflues et donc aucune phase de filtrage des informations observées n'est nécessaire.
- Extensibilité de la solution d'évaluation : nous rappelons que les sondes logicielles que nous utilisons pour l'observation sont composées de rubriques, chacune de ces rubriques se charge de détecter un type d'évènement bien déterminé et de tracer les informations relatives à cet évènement. Il est parfaitement possible et facile d'ajouter de nouvelles rubriques aux sondes pour observer et tracer de nouveaux types d'évènements selon les caractéristiques à évaluer. Ceci est principalement dû à la nature des aspects qui favorisent la modularité et facilitent l'extensibilité et l'évolution fonctionnelle du module.
- Limitation de l'interférence entre système d'évaluation et système évalué : un des apports majeurs de la programmation par aspects est la séparation des préoccupations, nous avons tiré profit de cet avantage pour assurer au mieux la séparation entre l'application évaluée et le module d'observation puisque c'est ce dernier qui est en interférence directe avec le SMA étudié.
- Performance : l'effet intrusif de l'observation peut engendrer une baisse des performances de l'application en termes de temps d'exécution et de consommation de la mémoire, car les entités chargées de l'observation partagent les mêmes ressources que les entités observées. Nous avons donc étudié la variation de ces deux critères généraux en prenant les mesures associées sur l'application en mode de fonctionnement normal, puis en activant le mécanisme d'observation. Les résultats des tests effectués à cet effet montrent que la surcharge en termes d'utilisation du temps CPU est de 1.7% en moyenne, et que la surcharge en termes de consommation de la mémoire est de 0,92% en moyenne. Ces deux valeurs sont considérées comme très acceptables puisqu'elles ne dépassent pas les 5% [Jai91].

4.2. Limites

Comme tout travail de recherche, le présent travail est sujet à extensions et améliorations. En effet dans cette thèse, nous nous sommes intéressés à l'analyse d'un aspect particulier des SMA à savoir l'aspect structurel. Deux caractéristiques ont été évaluées à cet effet. La première amélioration possible consiste à couvrir entièrement cet aspect par l'évaluation des autres caractéristiques de la même catégorie et d'établir les corrélations qu'il peut y avoir entres ces dernières.

Il serait également très intéressant d'aborder les autres facettes des SMA, notamment comportementale et d'interfaçage, dans le but d'aboutir à une solution d'évaluation complète. Dans ce cadre, nous étudierons l'adéquation de la modélisation par graphes à d'autres caractéristiques et son enrichissement éventuel.

Conclusion

Dans ce chapitre, nous avons présenté les principaux résultats d'évaluation de deux applications multi-agents. La première est une application multi-agents pour le diagnostic des pannes qui se décline en deux versions, une première version sans gestion des conflits éventuels et une deuxième avec gestion des conflits par arbitrage. La seconde application est une application de gestion de la production et de pilotage des chaines logistiques qui se décline en trois versions en fonction de la nature de l'architecture de pilotage utilisée, distribuée, centralisée ou mixte. Pour chacune de ces applications les différentes mesures associées aux caractéristiques de communication et d'organisation ont été générées et interprétées.

CONCLUSION ET PERSPECTIVES

Dans cette thèse nous nous sommes intéressés à la problématique de l'évaluation des performances, une question qui demeurait encore ouverte dans le cadre des systèmes multi-agents.

Nous avons commencé par une étude de l'état de l'art. Cette dernière a révélé que les travaux réalisés autour de cette thématique se déclinent essentiellement en cinq catégories, qui sont les suivantes :

- L'évaluation de la technologie agent en tant que nouveau paradigme du génie logiciel et sa comparaison aux technologies existantes telles que l'orientée-objet ;
- L'évaluation et la comparaison des approches de modélisation et des méthodologies de conception orientées-agent ;
- L'évaluation des outils de développement et des plates-formes d'exécution orientés-agent ;
- La validation et la vérification de modèles multi-agents ;
- L'évaluation des applications et des systèmes multi-agents développés.

L'étude de l'état de l'art nous a permis de souligner le manque considérable de travaux dans la dernière catégorie, et nous a aidés à nous positionner par rapport à la littérature.

Au cours de cette thèse, nous avons apporté plusieurs contributions.

Nous avons d'abord proposé une architecture d'évaluation des SMA qui s'articule autour de trois modules essentiels :

- Un module d'observation qui permet d'observer le comportement du système multi-agents évalué, de collecter les informations pertinentes sur les états et les évènements significatifs et de générer les traces d'exécution en vue de leur analyse ultérieure.

- Un module de modélisation qui permet de générer automatiquement le graphe représentatif du système multi-agents étudié en exploitant les traces d'exécution produites par le module d'observation.
- Un module de mesure qui permet d'analyser le graphe obtenu et de calculer les mesures associées aux différentes caractéristiques évaluées en s'appuyant sur la théorie des graphes. Les mesures ainsi obtenues sont ensuite interprétées.

Dans un premier temps, nous nous sommes concentrés sur le premier module et nous avons montré que l'observation est une étape indispensable pour pouvoir évaluer un SMA, puisque c'est le moyen qui nous a permis de collecter les informations nécessaires à son analyse. La technique d'observation que nous avons proposée se base sur la programmation orientée-aspects. En s'appuyant sur cette technologie très intéressante, nous avons défini des sondes d'observation logicielles dont le but était de tracer tous les évènements significatifs dans l'exécution d'un SMA.

Nous avons ensuite proposé une approche de mesure des SMA basée sur la modélisation. Pour ce faire nous avons commencé par identifier les caractéristiques fonctionnelles des SMA et nous les avons classifiées afin de pouvoir choisir un moyen de modélisation adéquat. Notre choix a porté sur les caractéristiques structurelles des SMA et pour les évaluer nous avons montré que le choix des graphes était le meilleur moyen de caractériser de telles propriétés. Ainsi, après avoir défini le modèle adopté, en nous référant à la théorie des graphes, nous avons associé plusieurs mesures à chacun des critères sélectionnés.

Afin de tester et valider notre approche d'évaluation, nous avons utilisé une application existante de diagnostic des pannes [Sad07], mais nous avons également développé une application multi-agents par nous-mêmes. C'est une application pour la gestion de la production et le pilotage des chaînes logistiques. Trois architectures de pilotages ont été mises en œuvre : le pilotage centralisé, le pilotage distribué et le pilotage mixte. Le modèle multi-agents proposé est un modèle générique et extensible qui permet la mise en œuvre de diverses configurations de chaînes logistiques possibles, l'intérêt étant de disposer d'une base expérimentale de test qui permette de valider le système d'évaluation proposé.

Les perspectives possibles de cette thèse sont multiples, la première consiste à explorer en profondeur l'aspect dynamique des SMA et d'étudier la variation des critères mesurés dans le temps ce qui permettrait d'ajouter une autre dimension aux interprétations possibles des résultats obtenus. Une autre extension intéressante à ce travail consiste à se

focaliser sur d'autres catégories de caractéristiques des SMA, telles que l'aspect comportemental ou d'interfaçage et de définir éventuellement d'autres moyens plus appropriés pour l'évaluation des caractéristiques qui s'y rattachent. A terme, il serait intéressant également d'aboutir à un système d'évaluation complet comportant toutes les caractéristiques fonctionnelles possibles des SMA et d'intégrer la phase d'évaluation au processus complet du cycle de vie afin de fournir aux concepteurs et aux développeurs un moyen d'analyse qui accompagne les différentes phases de mise en ouvre d'un SMA, ce qui permettra de tenir compte dès le départ de l'aspect très important de la performance.

BIBLIOGRAPHIE

[Akb08] Akbari, Z. O., & Faraahi, A. (2008). Evaluation framework for agent-oriented methodologies. *World Academy of Science, Engineering and Technology*, *45*, 418-423.

[Akb10] Akbari, O. Z. (2010). A survey of agent-oriented software engineering paradigm: Towards its industrial acceptance. *Journal of Computer Engineering Research*, *1*(2), 14-28.

[Amb06] Amblard, F., Rouchier, J., & Bommel, P. (2006). Evaluation et validation de modèles multi-agents. *Modélisation et simulation multi-agents. Applications pour les Sciences de l'Homme et de la Société, Hermès*, 103-140.

[Ami04] Amiguet, M., Nagy, A., & Baez, J. (2004). Towards an aspect-oriented approach of multi-agent programming. In *Proceedings of the 3rd Workshop on Modelling of Objects Components and Agents* (pp. 131-148).

[Arl04] Arlabosse, F., Gleizes, M. P., & Occello, M. (2004). Méthodes de Conception de Systèmes Multi-agents. *Systèmes multi-agents / Observatoire français des techniques avancées, ARAGO 29*.

[Bab05] Babczyński, T., Kruczkiewicz, Z., & Magott, J. (2005, September). Performance comparison of multi-agent systems. In *Proceedings of the 4th international Central and Eastern European Conference on Multi-Agent Systems and Applications* (pp. 612-615). Springer.

[Bar06] Barrat, A., Barthélemy, M., & Vespignani, A. (2006). Réseaux complexes et physique statistique. *Images de la Physique*, 47-53.

[Ben08a] Ben Hmida, F., Lejouad Chaari, W., & Tagina, M. (2008, March). Performance evaluation of multiagent systems: communication criterion. In *Proceedings of the 2nd KES International Conference on Agent and Multi-Agent Systems: Technologies and Applications* (pp. 773-782). Springer.

[Ben08b] Ben Hmida, F., Lejouad Chaari, W., & Tagina, M. (2008). Aspect-based multiagent systems observation for performance evaluation. In *Proceedings of the 4th International Conference on Intelligent Systems and Agents* (pp. 172-176). IADIS.

[Ben11] Ben Hmida, F., Lejouad Chaari, W., & Tagina, M. (2011). Graph theory to evaluate communication in industrial multiagent systems. *International Journal of Intelligent Information and Database Systems*, *5*(4), 361-388.

[Ben12a] Ben Hmida, F. B., Séguy, A., & Dupas, R. (2012). MultiAgent Systems for Production Planning and Control in Supply Chains. In *Proceedings of the 9th International Conference on Distributed Computing and Artificial Intelligence* (pp. 205-212). Springer.

[Ben12b] Ben Hmida, F. B., Séguy, A., & Dupas, R. (2012). MultiAgent Simulation and Evaluation of Supply Chain Control Architectures. In *proceedings of the 14th IFAC Symposium on Information Control Problems in Manufacturing* (pp. 781-786).

[Ben12c] Ben Hmida, F., Lejouad Chaari, W., Dupas, R. & Séguy, A. (2012). Graph-based evaluation of organization in multiagent systems. In *Proceedings of the 9th International Conference on Modeling, Optimization and SIMulation, Bordeaux.*

[Boi02] Occello, M., Guessoum, Z., & Boissier, O. (2002). Un essai de définition de critères pour l étude comparative de plates-formes multi-agents. *Technique et Science Informatiques (TSI)*, *21*(4), 549-552.

[Boi04] Boissier, O., Gitton, S., & Glize, P. (2004). Caractéristiques des Systèmes et des Applications. *Systèmes multi-agents / Observatoire français des techniques avancées, ARAGO 29*.

[Bom09] Bommel, P. (2009). *Définition d'un cadre méthodologique pour la conception de modèles multi-agents adaptée à la gestion des ressources renouvelables*. Thèse de doctorat, Université Sciences et Techniques du Languedoc, Montpellier II.

[Bon08] Bonnet, G., & Tessier, C. (2008). Evaluer un système multiagent physique-Retour sur expérience. In *Journées Francophones sur les Systèmes Multi-Agents* (pp. 13-22).

[Boo92] Booch, G. (1992). *Conception orientée objets et applications*. Addison-Wesley.

[Bou07] Bouabdallah, B. S., Saddem, R., & Tagina, M. (2007). A multiagent architecture for fault detection and isolation. In *Proceedings of the 1st International Workshop on Applications with Artificial Intelligence. Patras*.

[Bou01] Bouraqadi-Saâdani, N. M., & Ledoux, T. (2001). Le point sur la programmation par aspects. *Technique et Sciences Informatiques*, *20*(4), 505-528.

[Bou11] Boussebough, I. (2011). Les *systèmes multi-agents dynamiquement adaptables*. Thèse de doctorat, Université Mentouri, Constantine.

[Bou13] Bouzouita, K. (2013). *Évaluation des performances des agents rationnels*. Thèse de doctorat non publiée, École Nationale des Sciences de l'Informatique, Université de La Manouba, Tunis.

[Cam03] Campagne, J. C., & Cardon, A. (2003). Artificial emotions for robots using massive multi-agent systems. In *Proceedings of the International Conference on Social Intelligence Design, London.*

[Cam05] Campagne, J. C. (2005). *Systèmes multi-agents et morphologie*. Thèse de doctorat, Université Pierre et Marie Curie, Paris.

[Car04] Cardon, A. (2003). *Modéliser et concevoir une machine pensante: approche constructible de la conscience artificielle*. Collection Automates Intelligents, Paris: Vuibert.

[Car05] Cardon, A. (2005). *La complexité organisée: systèmes adaptatifs et champ organisationnel*. Paris: Hermès.

[Cer02] Cernuzzi, L., Rossi, G., & Plata, L. (2002, November). On the evaluation of agent oriented modeling methods. In *Proceedings of the 1st Workshop on Agent Oriented Methodology* (pp. 21-30).

[Cha01] Chaib-Draa, B., Jarras, I., & Moulin, B. (2001). Systèmes multi-agents: principes généraux et applications. *Edition Hermès*. Dans *Principes et architecture des systèmes multi-agents.* Paris: Hermès.

[Dij82] Dijkstra, E. W. (1982). On the role of scientific thought. In *Selected Writings on Computing: A Personal Perspective* (pp. 60-66). Springer.

[Duf06] Dufrêne G., Morvan S., *La programmation orientée aspects : AspectJ et JAC*. Rapport de Conception d'Applications Réparties, Mastère TIIR. USTL, Lille.

[Ela08] Elamy, A. H. H., & Far, B. (2008). On the evaluation of agent-oriented software engineering methodologies: a statistical approach. In *Agent-Oriented Information Systems IV* (pp. 105-122). Springer.

[Elf98] El Fallah Seghrouchni, A., Haddad, S., & Mazouzi, H. (1998). Etude des interactions basée sur l'observation répartie dans un système multi-agents. In *Journées Francophones sur les Systèmes Multi-Agents, Paris.*

[Fer95] Ferber, J., & Perrot, J. F. (1995). *Les systèmes multi-agents: vers une intelligence collective* (Vol. 16). Paris: InterEditions.

[Fer98] Ferber, J., & Gutknecht, O. (1998). A meta-model for the analysis and design of organizations in multi-agent systems. In *Proceedings of the International Conference on Multi Agent Systems* (pp. 128-135). IEEE.

[Fid96] Fidge, C. (1996). Fundamentals of distributed system observation. *IEEE Software*, *13*(6), 77-83.

[Fra07] Francois, J. (2007). *Planification des chaînes logistiques: Modélisation du système décisionnel et performance*. Thèse de doctorat, Université Sciences et Technologies, Bordeaux I.

[Gar01] Garcia, A., Silva, V., Chavez, C., & Lucena, C. (2002). Engineering multi-agent systems with aspects and patterns. *Journal of the Brazilian Computer Society*, *8*(1), 57-72.

[Gar06] Garcia, A., Chavez, C., & Choren, R. (2006). Enhancing agent-oriented models with aspects. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and MultiAgent Systems* (pp. 1332-1334). ACM.

[Gar02] Garneau, T., & Delisle, S. (2002). Programmation orientée-agent: évaluation comparative d'outils et environnements. In *Journées Francophones sur les Systèmes Multi-Agents* (pp. 111-124).

[Gau07] Gaud, N., Gechter, F., Galland, S., & Koukam, A. (2007). Holonic multiagent multilevel simulation application to real-time pedestrians simulation in urban environment. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence* (pp. 1275-1280). Morgan Kaufmann Publishers.

[Gna05] Gnanasambandam, N., Lee, S., & Kumara, S. R. (2005, July). An autonomous performance control framework for distributed multi-agent systems: a queueing theory based approach. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and MultiAgent Systems* (pp. 1313-1314). ACM.

[Gon95] Gondran, M., & Minoux, M. (1995). *Graphes et algorithmes*. Paris: Eyrolles.

[Gue04] Guessoum, Z., Briot, J. P., Faci, N., & Marin, O. (2004). Un mécanisme de réplication adaptative pour des SMA tolérants aux pannes. In *Journées Francophones sur les Systèmes Multi-Agents* (pp. 135-148).

[Hel03] Helsinger, A., Lazarus, R., Wright, W., & Zinky, J. (2003, July). Tools and techniques for performance measurement of large distributed multiagent systems. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent Systems* (pp. 843-850). ACM.

[Hor05] Horling, B., & Lesser, V. (2004). A survey of multi-agent organizational paradigms. *The Knowledge Engineering Review*, *19*(4), 281-316.

[Hub04] Hu, B., Liu, J., & Jin, X. (2004). From Local Behaviors to Global Performance in a Multi-Agent System. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology* (pp. 309-315). IEEE.

[Jai91] Jain, R. (1991). *The art of computer systems performance analysis : techniques for experimental design, measurement, simulation, and modeling*. Chichester: John Wiley & Sons.

[Jar97] Jard, C. (1997). Observation d'exécutions réparties: détection de propriétés d'ordonnancement. *Technique et science informatique*, *16*(2), 225-240.

[Jen00] Jennings, N. R. (2000). On agent-based software engineering. *Artificial intelligence*, *117*(2), 277-296.

[Jou08] Joumaa, H., Demazeau, Y., & Vincent, J. M. (2008). Evaluation of multi-agent systems: The case of interaction. In *Proceedings of the 3rd International Conference on Information and Communication Technologies: From Theory to Applications,* (pp. 1-6). IEEE.

[Jun04] Juneidi, S., & Vouros, G. (2004). Evaluation of Agent Oriented Software Engineering Main Approaches. In *Proceedings of the IASTED International Conference on, Software Engineering, Innsbruck*.

[Jur06] Jurasovic, K., Jezic, G., & Kusek, M. A Performance Analysis of Multi-Agent Systems. *International Transactions on Systems Science and Applications*, *1*(4).

[Kad09] Kaddoum, E., Gleizes, M. P., George, J. P., Glize, P., & Picard, G. (2009). Analyse des critères d'évaluation de systèmes multi-agents adaptatifs. In *Journées Francophones sur les Systèmes Multi-Agents* (pp. 123-132).

[Kan63] Kansky, K. J. (1963). *Structure of transportation networks: relationships between network geometry and regional characteristics*. Department of geography, Chicago University, Chicago.

[Kic01] Kiczales, G., Hilsdale, E., Hugunin, J., Kersten, M., Palm, J., & Griswold, W. (2001). Getting started with AspectJ. *Communications of the ACM*, *44*(10), 59-65.

[Kic97] Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J. M., & Irwin, J. (1997). Aspect-oriented programming. In *Proceedings of the 11th European Conference on Object-Oriented Programming* (pp 220-242). Springer.

[Kra94] Krackhardt, D. (1994). Graph theoretical dimensions of informal organizations. *Computational organization theory*, *89*(112), 123-140.

[Lad02] Laddad, R. (2002). *Separate software concerns with aspect-oriented programming*. Repéré à http://www.javaworld.com/javaworld/jw-01-2002/jw-0118-aspect.html.

[Lam11] Lamarche-Perrin, R., Demazeau, Y., & Vincent, J. M. (2011). *Macroscopic Observation of Multiagent Systems*. Rapport de recherche (RR-LIG-010). Laboratoire d'Informatique de Grenoble.

[Lam78] Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, *21*(7), 558-565.

[Leb10] Le Boudec, J. Y. (2010). *Performance evaluation of computer and communication systems*. EPFL Press.

[Lee98] Lee, L. C., Nwana, H. S., Ndumu, D. T., & De Wilde, P. (1998). The stability, scalability and performance of multi-agent systems. *BT Technology Journal*, *16*(3), 94-103.

[Leg03] Legras, F. (2003). *Organisation dynamique d'équipes d'engins autonomes par écoute flottante*. Thèse de doctorat, Ecole nationale supérieure de l'aéronautique et de l'espace, Toulouse.

[Lep00] Le Page, C., Bousquet, F., Bakam, I., Bah, A., & Baron, C. (2000). CORMAS: A multiagent simulation toolkit to model natural and social dynamics at multiple scales. In *Proceedings of the Workshop The ecology of scales, Wageningen.*

[Les04] Leszczyna, R. (2004). *Evaluation of agent platforms*. Rapport technique, Institute for the Protection and Security of the Citizen, Ispra.

[Lin00] Lind, J. (2001). Issues in agent-oriented software engineering. In *Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering* (pp. 45-58). Springer.

[Lin07] Lin, C. E., Kavi, K. M., Sheldon, F. T., Daley, K. M., & Abercrombie, R. K. (2007). A methodology to evaluate agent oriented software engineering techniques.

In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences* (pp. 60-60). IEEE.

[Lou08] Louie, M. A., & Carley, K. M. (2008). Balancing the criticisms: Validating multi-agent models of social systems. *Simulation Modelling Practice and Theory*,*16*(2), 242-256.

[Mae87] Maes, P. (1987). Concepts and experiments in computational reflection. *ACM Sigplan Notices*, *22*(12), 147-155.

[Mal09] Malek, M. (2009). *Introduction à l'analyse des réseaux sociaux*. Rapport de recherche, École internationale des sciences du traitement de l'information, Cergy.

[Man03] Manson, S. M. (2003). Validation and verification of multi-agent models for ecosystem management. In *Complexity and Ecosystem Management: The Theory and Practice of Multi-Agent Approaches*. Massachusetts: Edward Elgar Publishers, pp. 63-74.

[Man93] Mansouri-Samani, M., & Sloman, M. (1993). Monitoring distributed systems. *IEEE Network*, *7*(6), 20-30.

[Mei07] Meignan, D., Simonin, O., & Koukam, A. (2007). Simulation and evaluation of urban bus-networks using a multiagent approach. *Simulation Modelling Practice and Theory*, *15*(6), 659-671.

[Mul06] Mulet, L., Such, J. M., & Alberola, J. M. (2006). Performance evaluation of open-source multiagent platforms. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and MultiAgent Systems* (pp. 1107-1109). ACM.

[Myl02] Mylopoulos, J., Kolp, M., & Giorgini, P. (2002). Agent-Oriented Software Development. In *Proceedings of the 2nd Hellenic Conference on AI: Methods and Applications of Artificial Intelligence* (pp. 3-17). Springer.

[Nav12] Navarro, L., Corruble, V., Flacher, F., & Zucker, J. D. (2012). Mesoscopic Level: A New Representation Level for Large Scale Agent-Based Simulations. In *Proceedings of the 4th International Conference on Advances in System Simulation* (pp. 68-73).

[Ngu02] Nguyen, G., Dang, T. T., Hluchy, L., Balogh, Z., Laclavik, M., & Budinska, I. (2002). *Agent platform evaluation and comparison*. Rapport technique (Pellucid 5FP IST-2001-34519). Institute of Informatics, Slovak Academy of Sciences.

[Oma00] O'Malley, S. A., Self, A. L., & Deloach, S. A. (2000). Comparing performance of static versus mobile multiagent systems. In *Proceedings of the 2000 IEEE National Aerospace and Electronics Conference* (pp. 282-289). IEEE.

[Ott99] Ottogalli, F. G., & Vincent, J. M. (1999). Mise en cohérence et analyse de traces logicielles multi-niveaux. *Calculateurs parallèles*, *11*(2), 211-227.

[Paw04] Pawlak, R., Retaillé, J. P., & Seinturier, L. (2007). *Programmation orientée aspect pour Java/J2EE*. Paris: Eyrolles.

[Pet01] Petrie, C. (2001). Agent-based software engineering. In *Proceedings of the 1st International Workshop Agent-Oriented Software Engineering* (pp. 59-75). Springer.

[Pic04] Picard, G. (2004). *Méthodologie de développement de systèmes multi-agents adaptatifs et conception de logiciels à fonctionnalité émergente*. Thèse de doctorat, Université Paul Sabatier, Toulouse.

[Ric00] Ricordel, P. M., & Demazeau, Y. (2000). From analysis to deployment: A multi-agent platform survey. In *Engineering Societies in the Agents World* (pp. 93-105). Springer.

[Rob04] Robbes, R., Bouraqadi, N., & Stinckwich, S. (2004). Un modèle multi-agent unifiant les notions de groupe et d'aspect. In *Journées Francophones sur les Systèmes Multi-Agents* (pp. 105-118).

[Sab02] Sabas, A., Delisle, S., & Badri, M. (2002). A comparative analysis of multiagent system development methodologies: Towards a unified approach. *Cybernetics and Systems*, 599-604.

[Sad07] Saddam R. (2007) *Diagnostic des Systèmes à base d'agents*. Rapport de master, École Nationale des Sciences de l'Informatique, Université de La Manouba, Tunis.

[Sha04] Shakshuki, E., & Jun, Y. (2004). Multi-agent development toolkits: an evaluation. In *Proceedings of the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* (pp. 209-218). Springer.

[She01] Shehory, O., & Sturm, A. (2001). Evaluation of Modeling Techniques for Agent-Based Systems. In *Proceedings of the 5th International Conference on Autonomous Agents* (pp. 624-631). ACM.

[Sho90] Shoham, Y. (1990). *Agent-Oriented Programming*. Rapport de recherche (STAN-CS-1335-90), Computer Science Department, Stanford University, Stanford.

[Sho93] Shoham, Y. (1993). Agent-oriented programming. *Artificial intelligence*, *60*(1), 51-92.

[Sma03] Smacchia, P., & Vaucouleur, S. (2003). *Dossier Spécial AOP, Intérêts et Usages*. Repéré à http://www.vaucouleur.com/smacchia2003.pdf.

[Smi84] Smith, B. C. (1984). Reflection and semantics in Lisp. In *Proceedings of the 11th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (pp. 23-35). ACM.

[Stu03] Sturm, A., & Shehory, O. (2003). A framework for evaluating agent-oriented methodologies. In *Proceedings of the 5th Workshop on Agent-Oriented Information Systems* (pp. 94-109). Springer.

[Sud04] Sudeikat, J., Braubach, L., Pokahr, A., & Lamersdorf, W. (2005). Evaluation of agent–oriented software methodologies–examination of the gap between modeling and platform. In *Proceedings of the 5th International Workshop on Agent-Oriented Software Engineering* (pp. 126-141). Springer.

[Tho07] Thomas, V., Bourjot, C., & Chevrier, V. (2007). Construction de systèmes multi-agents par apprentissage collectif à base d'interactions. *Revue d'Intelligence Artificielle*, *21*(5-6), 643-672.

[Tri07] Trillo, R., Ilarri, S., & Mena, E. (2007). Comparison and performance evaluation of mobile agent platforms. In *Proceedings of the 3rd International Conference on Autonomic and Autonomous Systems*. (pp. 41-41). IEEE.

[Tve01] Tveit, A. (2001, May). A survey of agent-oriented software engineering. In *NTNU Computer Science Graduate Student Conference, Norwegian University of Science and Technology*.

[Wil05] Willig, A. (2005). *Performance Evaluation Techniques, Fundamentals and Big Picture*. Rapport de recherche, Telecommunication Networks Group, Technical University, Berlin.

[Woo00] Wooldridge, M., & Ciancarini, P. (2001). Agent-oriented software engineering: the state of the art. In *Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering* (pp. 1-28). Springer.

[Woo02] Wooldridge, M. (2002). *An Introduction to MultiAgent Systems*. London: John Wiley & Sons.

[Zöl06] Zöller, A., Braubach, L., Pokahr, A., Rothlauf, F., Paulussen, T. O., Lamersdorf, W., & Heinzl, A. (2006). Evaluation of a Multi-Agent System for Hospital Patient Scheduling. *International Transactions on Systems Science and Applications*, *1*(4), 375-380.