N° d'ordre :

Thèse

Présentée à

L'Université d'Evry-Val d'Essonne

Par

Narjes Khézami

Pour obtenir le diplôme de

Docteur de L'Université d'Evry-Val d'Essonne

Spécialité : **Robotique**

Vers un collecticiel basé sur un formalisme multi-agent destiné à la téléopération collaborative via Internet

---

Soutenue le 13 Décembre 2005

Devant le jury composé de :

B. David	Professeur, Ecole Centrale de Lyon	Rapporteur
P. Le Parc	Maître de conférences HDR, Université de Brest	Rapporteur
P. Fuchs	Professeur, Ecole des Mines de Paris	Examinateur
S. Otmane	Maître de conférences, Université d'Evry	Examinateur
M. Mallem	Professeur, Université d'Evry	Directeur de thèse

Remerciements

Je voudrais exprimer ma gratitude et mes sincères remerciements à Bertrand DAVID et Philippe LE PARC, respectivement Professeur à l'Ecole Centrale de Lyon et Maître de conférences HDR à l'université de Brest qui m'ont fait l'honneur d'être les rapporteurs de ce mémoire et qui ont apporté une contribution critique à celui-ci. Philippe FUCHS Professeur à l'Ecole des Mines de Paris qui a accepté d'examiner mon travail.

Malik MALLEM, Professeur à l'université d'Evry-Val d'Essonne, pour la confiance qu'il m'a accordée en m'accueillant dans son équipe ainsi que pour les conseils qu'il m'a apportés.

Et un très grand merci pour Samir OTMANE, Maître de conférences à l'université d'Evry d'avoir encadré ce travail et d'avoir corrigé ce manuscrit. Je suis également reconnaissante pour les conseils et les encouragements qu'il m'a dispensés et surtout de sa présence et de son soutien dans les moments difficiles.

Florent CHAVAND fera également l'objet de ma reconnaissance en tant qu'ancien directeur du laboratoire qui a mis à disposition sa structure pour m'accueillir lors du commencement de cette thèse. Son successeur à la tête du laboratoire, Étienne COLLE, sera tout autant remercié.

Je remercie également tous les doctorants pour l'ensemble des moments partagés ainsi que le personnel technique et administratif pour leur sympathie, leur grande disponibilité et leurs indispensables coups de main.

Je remercie également tous les stagiaires qui ont contribué directement ou indirectement à ce travail et plus particulièrement Alban ROCHEL pour sa sympathie, son encouragement, sa disponibilité et surtout pour son aide précieuse.

Enfin, je tiens à remercier ma famille pour le support et l'aide qu'elle m'a fournie et surtout au petit Rayan qui a supporté mon absence et parfois mon stress malgré son très jeune âge et qui m'a souvent aidé en me laissant travailler tranquillement le week-end.

La destination du chercheur dépend de la route qu'il suit.
[Ibn Al-'Arabi]

Table des matières

Liste des figures

La destination du chercheur dépend de la route qu'il suit.
[Ibn Al-'Arabi]

Introduction et Objectifs

Problématique et Objectifs

La téléopération désigne les principes et les techniques qui permettent à l'opérateur humain d'accomplir une tâche à distance, à l'aide d'un système robotique d'intervention (dispositif esclave), commandé à partir d'une station de contrôle (dispositif maître), par l'intermédiaire d'un canal de télécommunication.

Bien que la téléopération soit une technologie relativement jeune, elle a déjà subi un très grand nombre d'évolutions. Pendant les premières décennies de son histoire, les recherches ont porté sur la façon d'améliorer la performance et la fiabilité dans l'accomplissement des tâches en étudiant les lois de commande. Par la suite, la nécessité d'améliorer la sécurité des opérateurs a permis d'orienter les recherches sur les interactions homme-machine. Ceci a engendré de nombreux travaux notamment sur l'intégration de la réalité virtuelle et de la réalité augmentée dans les systèmes de téléopération.

Ces techniques (RA et RV) ont également permis de contourner le problème de retard inhérent à la téléopération. Par exemple, le fait d'augmenter l'image vidéo avec un modèle filaire, facilite la visualisation du monde 3D. Si l'opérateur désire effectuer un déplacement sur un robot réel, il peut le réaliser sur un robot virtuel qui n'est rien d'autre qu'un rehaussement graphique du vrai robot. L'opérateur peut alors décider de l'exécution de la tâche après avoir vu le résultat de la simulation, ainsi les problèmes d'une opération de télérobotique liés à la distance séparant les deux sites sont éliminés (problème de délai, $\cdots$).

Ces dernières années sont caractérisées par le besoin de réaliser et de concevoir des systèmes de téléopération interactifs multi-utilisateurs. Le travail collaboratif qui est un domaine de recherche pluridisciplinaire a cependant pris un essor considérable grâce à l'évolution de l'informatique et des supports de télécommunication. Principalement l'utilisation du concept des multi-agents a permis de modéliser le comportement très complexe des tâches collaboratives.

Notre travail de recherche s'inscrit dans le cadre du Travail Collaboratif Assisté par Ordinateur (TCAO) qui se traduit par l'assistance d'un groupe d'utilisateurs dans leur travail de collaboration. L'objectif de cette recherche est de modéliser, de concevoir, d'implémenter et d'évaluer un système de collaboration. Ce dernier doit être capable de prendre en charge la collaboration de plusieurs utilisateurs distants pour préparer et réaliser des missions de téléopération. Dans le souci d'assister le développement d'un tel système, nous proposons des formalismes et des notations qui seront utilisés pour spécifier le comportement désirable du système de collaboration. Une exigence est que nous devrions être capables de transférer les spécifications de ce système à l'implémentation.

A fin de présenter les travaux réalisés nous proposons une organisation du manuscrit en cinq chapitres que nous résumons ci-dessous.

Le premier chapitre a comme objectif principal la recherche des concepts, des méthodes et des outils nécessaires pour répondre à notre problématique. Pour cela nous avons abordé deux domaines de recherche qui sont les Systèmes Multi-Agents (SMA) et le Travail Collaboratif Assisté par Ordinateur (TCAO). Les études menées dans le domaine des SMA ont deux objectifs. Le premier consiste à présenter les différents concepts, propriétés et formalismes des SMA utiles pour la modélisation des systèmes complexes et distribués. Le second objectif vise à établir un bilan des plates-formes de développement multi-agents pouvant être utilisées pour l'implémentation de tels systèmes. Dans le domaine du TCAO, l'étude présentée a comme objectifs de tenter de répondre aux questions suivantes : Quelle définition peut-on attribuer au terme collaboration? Comment peut-on modéliser la collaboration et quels sont les modèles d'architectures existants? Nous terminons ce chapitre, par un bilan à partir duquel nous dégagerons quelques méthodes et outils essentiels pour la réalisation d'un SMA pour la Collaboration (SMA-C).

Le second chapitre aborde la modélisation du SMA-C. Dans la première partie, nous présentons une vue d'ensemble des recherches effectuées sur les SMA appliqués au TCAO. Nous introduisons aussi la notion d'agent collaboratif et quelques applications dans ce domaine. Dans la deuxième partie de ce chapitre nous proposons un formalisme pour la collaboration utile pour la spécification de la collaboration. La troisième partie est dédiée à la modélisation du SMA-C en présentant notre modèle d'agent collaborateur.

Le troisième chapitre est consacré à la conception, à l'implémentation et à l'évaluation du SMA-C. Il s'agit d'utiliser le modèle d'agent collaborateur proposé dans le second chapitre pour concevoir, implémenter et évaluer l'architecture logicielle du SMA-C. Dans la première partie de ce chapitre nous présentons les deux outils choisis pour la conception et l'implémentation de l'architecture logicielle du SMA-C. Le premier outil est l'UML (Unified Modeling Language), il est choisi pour la modélisation et la conception de l'architecture logicielle. Le second outil choisi est JADE (Java Agent DEvelopment framework), c'est une plateforme de développement d'agents en java utile pour l'implémentation de l'architecture logicielle du SMA-C. La seconde partie de ce chapitre aborde la conception et l'implémentation de l'architecture logicielle du SMA-C. Une dernière partie est dédiée à l'évaluation du SMA-C en proposant également un simulateur développé à l'occasion de cette évaluation.

Le quatrième chapitre a comme objectif de tester le SMA-C développé dans le troisième chapitre sur une application réelle avec des données et des contraintes réelles. Nous commençons ce chapitre par une présentation du domaine de la téléopération avec une présentation de quelques systèmes de téléopération via Internet existants. Dans la seconde partie de ce chapitre nous proposons de tester notre SMA-C sur le premier système en France de téléopération en réalité augmentée via Internet ARITI[1] (Augmented Reality Interface for Teleoperation via Internet) développé au LSC en 1998 et référencé sur le site web de la NASA[2] depuis janvier 2000. Il s'agit donc d'étendre le système ARITI avec de nouvelles fonctionnalités de collaboration en proposant une nouvelle version d'ARITI Collaboratif (ARITI-C). Nous présentons son architecture logicielle, son Interface Homme Machine (IHM) ainsi qu'une description des différentes missions et actions prises en compte pour la téléopération collaborative.

Dans le dernier chapitre nous présentons une évaluation de l'ergonomie de l'IHM de ARITI-C ainsi que les statistiques d'utilisation obtenues grâce à l'outil mis en ouvre pour l'analyse et l'évaluation de la collaboration. Nous commençons ce chapitre par une présentation des méthodes d'évaluation des systèmes collaboratifs en optant par la suite sur les tests d'utilisabilité. Dans la seconde partie nous présentons le protocole d'évaluation de l'ergonomie de l'IHM de ARITI-C ainsi que les résultats quantitatifs et qualitatifs obtenus. Dans la dernière partie de ce chapitre nous exposons l'outil d'analyse et d'évaluation de la collaboration développé ainsi que quelques statistiques d'utilisation obtenues.

[1] http://lsc.univ-evry.fr/Projets/ARITI/index.html
[2] http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html

Contributions

Nous résumons ci-dessous les principales contributions de cette thèse :

- Identification des besoins requis d'une plateforme générique pour la collaboration.
- Étude et comparaison des différentes approches de collaboration pour l'agent collaboratif.
- Proposition d'un formalisme pour la collaboration.
- Proposition d'un formalisme d'agent collaborateur.
- Proposition d'un simulateur pour l'évaluation des performances du SMA-C.
- Proposition d'un collecticiel pour la téléopération collaborative via Internet.
- Proposition d'un outil pour l'analyse et l'évaluation de la collaboration.

Conventions de notation

- Les noms de classes sont constitués de mots concaténés. Chaque mot est en minuscules à l'exception de la première lettre. Le caractère "_" est proscrit. Exemple : `MaPremiereClasse`.
- Les méthodes sont nommées de la même manière, à l'exception de la toute première lettre, en minuscules. Exemple : `trierLaListe()`.
- Les variables sont nommées comme les méthodes, le premier mot étant un suffixe définissant l'endroit de déclaration de la variable : membre `m`, paramètre `p` ou locale `a`. Exemple : `mUnMembre`.
- Les constantes (`final`) sont en majuscules, mots séparés par des "_". Exemple : `DIMENSION_DE_LA_FENETRE`.

Chapitre 1

Etat de l'art

1.1 Introduction

Qu'il s'agisse de diagnostic, de conception de systèmes, de planification, d'ordonnancement, etc., les problèmes réels sont souvent abordés par un groupe d'entités (personnes, capteurs, etc.). Chaque entité est chargée d'une tâche spécifique mais ne possède pas les ressources suffisantes pour aboutir, individuellement, à la résolution du problème traité. Ces entités plus au moins expertes et disposant d'une certaine autonomie entre elles dans leur environnement. Une décision se dégage de ces interactions et celle-ci est souvent meilleure que celle qui aurait été produite par une seule entité. Par conséquent, le recours aux systèmes multi-agents et l'idée de partager et distribuer entre plusieurs entités l'ensemble des connaissances, le raisonnement et la décision deviennent nécessaires. Chacune de ces entités est alors spécialisée dans un sous-domaine du domaine initial. La collaboration, la coopération et parfois le conflit entre les entités du système permettent d'améliorer le degré de contribution de chacune d'elle à la résolution du problème global.

Au travers de ce chapitre, nous passons en revue les différentes approches étudiées et mises en œuvre dans les deux domaines de recherche abordés, à savoir les Systèmes Multi-Agents (SMA) et le Travail Collaboratif Assisté par Ordinateur (TCAO). Dans la première partie, nous dressons un panorama des systèmes multi-agents afin d'identifier les concepts, les méthodes et les outils utilisés pour la modélisation et la réalisation des SMA. Dans la seconde partie, nous présentons d'une part, quelques notions et définitions nécessaires à la compréhension du domaine de TCAO et d'autre part, nous ferons un tour d'horizon des modèles d'architecture mis en œuvre pour la conception des systèmes de TCAO.

Nous terminerons ce chapitre par un bilan dans lequel nous dégagerons quelques problématiques, et en situant notre proposition.

1.2 Les Systèmes Multi-Agents : SMA

L'évolution des domaines d'application de l'Intelligence Artificielle (IA) pour recouvrir des domaines complexes et hétérogènes tels que l'aide à la décision, la reconnaissance et la compréhension des formes, etc., a montré les limites de l'approche classique de l'IA qui s'appuie sur une centralisation de l'expertise au sein d'un système unique.

Les travaux menés au début des années 70 sur la concurrence et la distribution ont contribué à la naissance d'une nouvelle discipline à savoir l'Intelligence Artificielle Distribuée (IAD)et la technologie des SMA est un des aspects de cette discipline. Son développement est une conséquence de la limitation de l'IA classique sur le plan de la structuration des connaissances, ce qui a nécessité de trouver des techniques performantes de modélisation et de simulation. Le développement de l'informatique et des systèmes distribués en particulier, a aussi favorisé l'extension des SMA vers des applications destinées au grand public.

Nous présenterons dans le paragraphe suivant l'entité appelée Agent qui est la pierre angulaire de tout SMA.

1.2.1 Le concept d'Agent

L'agent est l'entité élémentaire d'un SMA. Malgré l'accroissement des travaux dans ce domaine, les chercheurs n'ont pas pu trouvé un consensus sur la définition exacte de la notion d'agent.

Leonnec [LEO 96] a défini *l'agent comme un objet informatique (au sens des langages objet) dont le comportement peut être décrit par un script, qui dispose de ses propres moyens de calcul, et qui peut se déplacer de places en places pour communiquer avec d'autres agents.*

Certains chercheurs ont donné la définition d'agent à travers une bonne description du fonctionnement des agents. Un exemple d'une telle définition peut se trouver dans [COH 88], Cohen et Levesque considèrent qu'*un agent doit nécessairement être motivé par un but à atteindre sinon son existence dans son environnement n'aurait pas de sens. Il peut percevoir l'environnement mais peut n'en posséder qu'une représentation partielle et parfois même aucune. Il peut communiquer avec les autres agents de son environnement et doit avoir des compétences qui lui permettent d'atteindre ses objectifs.*

D'autres chercheurs ont considéré un agent comme une entité intelligente, agissant de façon rationnelle et intentionnelle par rapport à ses buts et à l'état courant de ses connaissances [DEM 91]. Cette définition a été complétée dans [SHO 93], en considérant que les agents peuvent être asservis. Ils agissent de façon

continue et autonome dans un environnement où des processus prennent place et où d'autres agents existent. En insistant sur le fait que cet environnement peut parfois être totalement inconnu. Un agent peut aussi se reproduire [SYC 98], cette propriété revêt toute son importance dans les nouvelles applications de travail coopératif à travers le Web.

Ferber [FER 95] définit un agent comme *une entité réelle ou virtuelle dotée de capacités d'action et de réaction à l'environnement dans lequel elle se situe.* Il considère aussi un agent comme *un objet ayant des capacités supplémentaires, à savoir recherches de satisfaction d'une part, et communications à base de langages plus évolués d'autre part* [FER 97].

De nombreux chercheurs tels que Wooldridge et Jennings [WOO 95][JEN 98], Nwana [NWA 96], Bradshaw [BRA 97], Shoham [SHO 97] ont affiné ultérieurement la définition d'agent. Bradshaw [BRA 97] a défini un agent comme étant *une entité logicielle qui fonctionne continuellement et de façon autonome dans un environnement particulier, souvent peuplé par d'autres agents et processus. Le besoin de continuité et d'autonomie fait que l'agent supporte des activités d'une manière intelligente et flexible et peut s'adapter à l'environnement sans requérir l'intervention humaine. Un agent qui fonctionne dans un environnement sur une longue période de temps doit être capable d'en retirer une expérience. Dans un environnement composé d'un ensemble d'agents, un agent est capable de coopérer et de communiquer, et éventuellement de migrer de plateforme en plateforme pour cela.*

1.2.2 Les propriétés d'agent

Indépendamment des définitions que nous avons énoncées, tout agent possède un ensemble de propriétés qui permet de le caractériser. Les principales caractéristiques des agents énumérées dans plusieurs travaux sont présentées dans cette section.

La modularité : A partir du constat qu'un seul mécanisme de raisonnement, un seul lieu de traitement, un seul recueil de connaissances étaient inappropriés dans de nombreuses applications, le problème de modularité a été posé. Par la suite, la prolifération des capacités de traitement et de communication n'ont fait que renforcer ce constat, avec la volonté d'interconnecter et de rendre accessibles en tout lieu et à tout moment des ressources informatiques existantes (données, informations, traitements), d'intégrer des systèmes existants, de les faire coopérer, interagir et collaborer avec un ou plusieurs utilisateurs.

L'autonomie : Un agent autonome est capable d'agir seul, c'est-à-dire de prendre des décisions selon des critères qui lui sont propres et sans l'intervention d'un autre agent ou d'un opérateur humain. Le fait que cette propriété soit une caractéristique des SMA renforce l'indépendance de l'exécution ou du comportement d'un agent vis-à-vis d'une invocation provenant d'une entité extérieure. Un agent se distingue des objets de programmation par objet, dans la mesure où ces derniers sont passifs et exécutent uniquement les ordres. L'autonomie d'un agent peut se définir par trois capacités :

- Les agents ont une existence propre, indépendante de l'existence des autres.
- Les agents sont capables de maintenir leur viabilité dans des environnements dynamiques sans contrôle extérieur.
- La prise de décision interne des agents est uniquement en fonction des perceptions, connaissances et représentations du monde propre aux agents.

La communication : Afin d'assurer un comportement global cohérent, les agents interagissent entre eux. L'une des voies possibles est une communication indirecte entre agents par le biais de l'environnement dans lequel ils sont plongés. Une autre possibilité est la communication directe entre les agents. Ces communications sont souvent modélisées par des communications élaborées, le plus souvent asynchrones. Répondant à la distribution aussi bien physique que fonctionnelle des agents, la communication dans un SMA est généralement conçue comme asynchrone, du fait des temps de communication allongés (délai d'acheminement réseau), du risque de perte de messages ou de changement de l'ordre d'arrivée de ces messages.

L'interaction : L'interaction des agents doit permettre au système de la façon la plus autonome possible de résoudre un problème. De même, cela doit permettre une meilleure réactivité aux situations nouvelles. Par conséquent, un agent peut agir sur le monde qui l'environne, c'est à dire sur les autres agents présents dans son univers et sur l'environnement lui-même. Cette intervention peut prendre la forme d'une modification de l'état des autres agents qu'il côtoie, que ce soit au niveau de leurs connaissances ou au niveau de leur activité. C'est l'ensemble de ces actions réalisables par un agent que l'on appelle interaction. L'interaction est une caractéristique très importante, comme l'affirme Ferber [FER 95] : "Un agent sans interaction avec d'autres agents n'est plus qu'un système de traitement d'information, dépourvu de caractéristiques adaptatives".

La coordination : Un SMA est une collection de points de vue locaux différents, voire conflictuels, qui établissent un ou plusieurs réseaux de relations

entre les agents. Les réseaux de relations existant entre ces agents sont complexes et multiples. Découlant des caractéristiques des agents, entités logicielles capables de manipuler différentes connaissances, ces relations peuvent porter sur les buts, les plans, les actions ou les ressources. Elles donnent lieu à des situations de coordination où plusieurs agents peuvent être impliqués. Cette coordination permet de répondre aux différents avantages, à savoir l'amélioration des gains des agents, la résolution des conflits, etc., si la situation de coordination dans laquelle se trouve l'agent est qualifiée de positive. Néanmoins, elle pose de nombreux problèmes de coordination si la situation est qualifiée de négative, i.e., en défaveur de l'agent (la diminution de ses gains).

La personnalisation : Un SMA doit adapter continuellement sa relation avec l'utilisateur, ce qui conduit à une personnalisation des comportements. Cette fonctionnalité permet de maintenir des informations sur les préférences de l'utilisateur et peut être construite explicitement ou déduite par le système en observant son comportement.

L'intelligence : L'intelligence, au sens large, est l'aptitude à comprendre et à s'adapter à une situation nouvelle. L'**intelligence d'un agent** est basée sur un agrégat de caractéristiques et concerne la capacité à apprendre, à communiquer et être autonome [NWA 96].

L'emergence : Des éléments perturbateurs (l'environnement, l'utilisateur ou d'autres agents) imposent au système de s'adapter au contexte et de se restructurer en permanence afin de conserver des performances acceptables. Cette caractéristique d'auto organisation n'est pas propre aux SMA, mais traverse beaucoup d'autres disciplines scientifiques. Elle correspond à l'émergence d'une cohérence globale à partir uniquement des interactions locales au niveau de composants initialement indépendants et les règles d'interactions sont exécutées sans aucune référence à la formation globale.

Il y a bien entendu d'autres propriétés que nous aurions pu rajouter comme l'ouverture, la décentralisation, l'intelligibilité ...Mais il semble que les propriétés que nous avons citées ci-dessus soient communes à toute la communauté de chercheurs en Intelligence Artificielle. Cependant, aucun produit à l'heure actuelle ne rassemble toutes ces caractéristiques, bien que beaucoup d'entreprises profitent de ce flou terminologique pour qualifier leur logiciel d'agent intelligent.

1.2.3 Vers le Multi-agent

Les SMA permettent de répartir un problème sur un certain nombre d'entités coopérantes. Elles permettent aussi de coordonner intelligemment le comportement de ces entités selon des lois sociales. Ces entités ou agents sont autonomes et interagissent dans un environnement pour la résolution de problème qui dépasse les capacités individuelles de chaque agent.

Plusieurs définitions ont été données à un système multi-agent. Durfee [DUR 89] a défini un système multi-agent comme *un réseau fortement couplé d'entités, qui travaillent ensemble pour solutionner des problèmes qui dépassent les capacités individuelles ou les connaissances de chaque entité.*

Une autre définition plus générale des SMA est donnée dans [JEN 98], il préconise que :

- *Chaque agent possède des capacités incomplètes pour résoudre le problème.*
- *Il n'y a pas de système de contrôle global.*
- *Les données sont décentralisées.*
- *Le calcul est asynchrone.*

En fonction de la taille d'un agent, de sa complexité, de ses connaissances et de son raisonnement, nous pouvons classer les approches multi-agents en trois grandes catégories : cognitive, réactive et hybride.

L'approche cognitive : Les concepteurs des SMA cognitifs s'inspirent du comportement humain pour définir la structure et le raisonnement d'un agent cognitif. Pour pouvoir anticiper ou expliquer ses actions et celles des autres agents, un agent cognitif doit posséder des connaissances sur les autres agents [BOR 94].

Généralement, les systèmes multi-agents cognitifs sont composés d'un petit nombre d'agents de grande granularité, i.e., ils possèdent un minimum de connaissances et des comportements de haut niveau leur permettant de s'organiser, de se regrouper, de coopérer, d'apprendre à coopérer en se servant de leurs expériences, et également de prévoir les résultats de leurs comportements. Les premières approches de conception de systèmes multi-agents cognitifs sont dues à Lenat [LEN 75], Hayes-Roth [HAY 79] et [HAY 85] qui ont introduit les architectures cognitives à base de tableaux noirs.

L'approche réactive : Plusieurs chercheurs ont travaillé sur les systèmes multi-agents réactifs [AGR 87], [BRO 91], [FER 95]. Ils se sont inspirés des phénomènes biologiques, ils observent les comportements organisationnels biologiques ou éthologiques et tentent de les reproduire par la suite. Les agents sont dotés d'un

modèle de comportement du style automate selon un modèle stimulus-réponse. Contrairement aux agents cognitifs, les sociétés d'agents réactifs sont composées d'un nombre considérable d'agent de faible granularité.

En général, un agent réactif ignore ses expériences passées car il ne possède pas de processus de raisonnement sophistiqués qui lui permettent de planifier ou d'apprendre. Ces agents sont très rapides dans la prise de décision car ils sont dotés d'un minimum de connaissances et de modèles de raisonnement à base de règles. Pour être fidèle aux modèles biologiques, les concepteurs d'agents réactifs évitent d'utiliser toute communication directe entre agents. Les agents ont alors recours à l'observation des changements qui affectent l'environnement à travers ses différents états, ce qui assure une transmission indirecte des connaissances entre agents.

L'approche hybride : Pour surmonter les faiblesses de chacune des deux approches précédentes, i.e., la difficulté de mise en œuvre de l'approche cognitive dans les environnements complexes et à forte évolution, et le manque de modèles formels dans l'approche réactive, plusieurs chercheurs se sont intéressés à l'approche hybride [BUR 92], [STE 93], [GUE 99]. Le postulat de cette approche est de maintenir une certaine réactivité de l'agent en le dotant de composants réactifs et de rendre les autres composants cognitifs pour garantir un raisonnement de qualité. Les fondateurs de cette approche argumentent son intérêt par les avantages qu'elle procure, notamment:

- un agent hybride a une structure modulaire, ce qui est concrètement recommandé dans le développement de tout processus artificiel pour garantir l'évolution et la maintenance du système ;
- les capacités de traitement d'un agent peuvent être améliorées car ces différents composants peuvent fonctionner simultanément ;
- le composant réactif de l'agent devient plus performant car l'organisation des connaissances d'un agent en partitions permet à chacun des composants réactifs ou cognitifs de manipuler partiellement ou totalement cette connaissance.

1.2.4 Modélisation et développement des Systèmes Multi-agents

Dans cette partie nous présentons quelques principes et concepts utilisés pour la modélisation des systèmes multi-agents ainsi que les plateformes de développement nécessaires pour leur mise en œuvre.

1.2.4.1 Aspect méthodologique

Une méthodologie doit permettre de faciliter le processus d'ingénierie des systèmes. Ces dernières années, le besoin de fournir des modèles, des méthodologies, des plateformes se fait sentir afin de faciliter la prise en compte de la complexité des systèmes à concevoir. Les deux grandes familles de travaux concernant les méthodes orientées agent étendent les concepts soit vers des méthodes orientées objet, soit vers des méthodes issues de l'ingénierie des connaissances.

D'après [ARL 04], une méthode est définie commme *"un processus rigoureux permettant de générer un ensemble de modèles qui décrivent divers aspects d'un logiciel en cours de développement en utilisant une certaine notation bien définie"*. Une méthode de développement de systèmes multi-agents est constituée d'un processus, d'une notation et d'outils pour servir de support à ce processus et à ces notations et/ou pour aider le développeur (CAME: Computer Aided Method Engineering tool).

[BOO 92] donne la définition suivante d'une méthodologie : *"une méthodologie est un ensemble de méthodes appliquées tout au long du cycle de développement d'un logiciel, ces méthodes étant unifiées par une certaine approche philosophique générale"*. Dans [ARL 04], une méthodologie est un ensemble de guides qui couvrent tout le cycle de vie du développement d'un logiciel : ces guides sont à la fois techniques et gèrent le projet.

La méthodologie doit donc amener le concepteur à réfléchir sur la décomposition de son système et doit donc l'aider à décider si un composant sera un agent ou un simple objet. Un agent possède un objectif individuel (fonction de satisfaction), des ressources, une représentation partielle de l'environnement, des compétences et il offre des services. Son comportement est fonction de ses observations, de ses connaissances, de ses croyances, de ses compétences et de ses interactions.

Les modèles manipulés lors du développement peuvent être classés en modèles d'entité, modèles de structure et modèles dynamiques [LIN 01]. Les démarches, comme dans les méthodes traditionnelles (démarches fondées sur les données, sur les traitements, etc.), peuvent privilégier certains de ces axes. Dans le domaine des systèmes multi-agents, on retrouve les démarches fondées sur :

- Les entités : les tâches dans le système DESIRE [BRA 99], les rôles dans GAIA [WOO 01] ou AALAADIN [GUL 98].
- La dynamique : les scénarios dans MASB [MOU 96], les buts dans MaSE [DEL 01].

- La structure : l'émergence dans la méthode CIRTA [MUL 98], [LAB 98] et dans la méthode ADELFE [PIC 04].

On peut aussi trouver des méthodes effectuant une utilisation conjointe des différentes démarches, comme dans VOYELLES [DEM 01] ou dans MASSIVE [LIN 01].

D'après certains chercheurs [JAC 99], [RIC 00], [CAS 00] et [ARL 04], une méthodologie doit tenir compte de tous les aspects du cycle de vie d'un logiciel à savoir :

- L'analyse de besoin d'un système multi-agent.
- La conception globale.
- La conception détaillée.
- Le développement.
- Le déploiement et la maintenance.

Dans ce qui suit, nous présentons un formalisme de système multi-agent conçu par Ferber, qui à notre avis, constitue une bonne base à partir de laquelle nous pouvons concevoir et implémenter des SMA.

1.2.4.2 Un formalisme pour les Systèmes Multi-agents

Les théories formelles d'agents sont les spécifications de l'agent, pas seulement dans le sens de fournir des descriptions et des contraintes sur le comportement de l'agent, mais aussi dans le sens du terme 'spécification' du génie logiciel. Cela veut dire que les théories formelles d'agent fournissent également une base à partir de laquelle nous pouvons concevoir, implémenter et vérifier les systèmes d'agents. Les agents sont un prochain pas naturel pour le génie logiciel; ils représentent une nouvelle façon fondamentale pour la considération des systèmes distribués complexes, en contenant des composants autonomes coopérants.

Les propriétés, identifiées en utilisant des formalismes, servent à mesurer et à évaluer les implémentations des systèmes d'agent. Quelques propriétés paraissent être non implémentable actuellement, parce qu'elles traitent un aspect idéalisé d'agencement, tel que la connaissance. Encore, ces propriétés peuvent être utiles parce que les approximer fournit une base de comparaison des implémentations. L'intérêt principal, bien sûr, est que le développement de théories formelles devrait être guidé par les besoins d'applications pratiques d'agents.

Ferber considère un agent en interaction avec le monde comme un système composé de deux sous-systèmes dynamiques couplés, le couplage s'effectuant au travers des perceptions que l'agent a du monde et des actions qui modifient ce

monde [FER 97]. Il représente un système multi-agent par le couple $< a, w >$ où a est un agent et w un monde qui sont chacun décrits ainsi :

$$\begin{aligned} a &= < P_a, Percept_a, F_a, Infl_a, S_a > \\ w &= < E, \Gamma, \Sigma, R > \end{aligned} \tag{1.1}$$

- P_a représente la fonction de perception de l'agent. C'est la fonction qui permet à un agent, étant dans un état, de discerner l'ensemble de données qui peuvent influencer son comportement,
- $Percept_a$ l'ensemble des stimuli et sensations qu'un agent peut recevoir,
- F_a la fonction de comportement de l'agent qui détermine l'état de l'agent à partir de ses perceptions et de son état précédent,
- $Infl_a$ la fonction d'action de l'agent, c'est-à-dire la fonction qui tend à modifier l'évolution du monde en produisant des influences [FER 96], et ce à partir d'un état interne à l'agent, il génère un ensemble d'influences,
- S_a l'ensemble des états internes de l'agent,
- E l'espace dans lequel l'agent évolue,
- Γ l'espace des influences produites par l'agent et ayant comme conséquences de modifier l'évolution du monde,
- Σ l'ensemble des états de l'agent,
- R la loi d'évolution du monde; un agent change d'état suite aux influences qu'il produit.
  Un agent réagit, donc, en fonction des actions des autres, et toutes ces actions/réactions changent l'évolution de cet agent dans son environnement :

$$\begin{aligned} P_a &: \Sigma \rightarrow Percept_a \\ Infl_a &: S_a \rightarrow \Gamma \\ F_a &: S_a \times Percept_a \rightarrow S_a \\ R &: \Sigma \times \Gamma \rightarrow \Sigma \end{aligned} \tag{1.2}$$

Ces fonctions satisfont les équations suivantes qui décrivent la dynamique de l'agent en interaction avec son environnement. L'état interne d'un agent évolue au cours du temps en conséquence des perceptions qu'il reçoive à l'instant t. Ainsi que son état global qui évolue au cours du temps en conséquence des influences qu'il produit au même instant t :

$$\begin{aligned} s_a(t+1) &= F_a(s_a(t), P_a(\sigma(t))) \\ \sigma(t+1) &= R(\sigma(t), Infl_a(s_a(t))) \end{aligned} \tag{1.3}$$

où s_a est un élément de S_a et
σ un élément de Σ.

Dans un système multi-agent, les différentes actions des agents sont combinées par l'intermédiaire d'un opérateur de combinaison d'influences Π qui prend les résultats des actions des agents et les combine de manière simple (union des influences, sommation vectorielle, etc.). Dans ce cas, un système multi-agent est défini par un triplet $< A, w, \Pi >$,

où A est un ensemble d'agents décrits comme précédemment,
w un monde et
Π un opérateur de combinaison d'influences.

La dynamique du système est alors donnée par le système des $n+1$ équations suivantes :

$$\begin{aligned} s_1(t+1) &= F_1(s_1(t), R(\sigma(t))) \\ &\cdots \\ s_n(t+1) &= F_n(s_n(t), R(\sigma(t))) \\ \sigma(t+1) &= R\left(\sigma(t), \prod_{i=1\cdots n} Infl_i(s_i(t))\right) \end{aligned} \tag{1.4}$$

Donc, pour un système multi-agent composé de n agents, l'état interne de chaque agent, à un instant $t+1$, dépend de son état interne à l'instant t et de son évolution dans son environnement, en suivant la même loi que suivent les autres agents du système. L'état global du système à l'instant $t+1$ évolue en fonction de l'état précédent à l'instant t et de la combinaison des différentes influences produites par tous les agents du système. Il est primordial dans un système multi-agent, que tous les agents suivent la même loi d'évolution, c'est-à-dire qu'ils ont les mêmes règles au sein du même groupe. De ce fait, la société d'agents repose sur une loi commune à tous les agents.

Pour faciliter l'implémentation des SMA modélisés, il est nécessaire d'utiliser des plateformes de développement adaptées. Dans ce qui suit nous présentons brièvement quelques plateformes de développement multi-agents pouvant être utilisées.

1.2.4.3 Plateformes multi-agents

La notion de plateforme est liée à l'implémentation des systèmes multi-agents et elle constitue un réceptacle au sein duquel les agents peuvent évoluer. Les plateformes sont un environnement permettant de gérer le cycle de vie des agents et dans lequel les agents ont accès à certains services. D'après [ARL 04], nous pouvons classer les plateformes en trois groupes :

- Les plateformes de simulation : elles servent à reproduire l'environnement ou le comportement d'un système complexe afin d'en étudier la dynamique, par exemple, la plateforme swarm[1].
- Les plateformes d'exécution : elles fournissent des outils d'implémentation basés sur des modèles particuliers, mais elles ne sont pas associées à des méthodes, par exemple, les plateformes Jack, JADE et Voyager.
- Les plateformes de développement : elles servent du support à une méthode en fournissant des outils pour asssiter une démarche de conception, par exemple, les plateformes AgentBuilder, Madkit, Volcano et ZEUS.

Dans ce qui suit, nous présentons brièvement quelques plateformes multi-agents.

Swarm du Swarm Development Group[1] :

Swarm est un environnement de simulation de société d'agents à grande échelle (i.e. en nombre d'agents) initialement développé au Sante Fe Institute. Dans Swarm une simulation est constituée d'un ensemble d'agents et d'un calendrier indiquant ce qu'un agent peut envoyer comme message et à quel moment. L'association des agents au calendrier définissant la dynamique des échanges constitue un modèle dans Swarm. Nous citons cette plateforme car c'est la plus reconnu dans le domaine de la vie artificielle. Il faut cependant bien noter que ce type de plateforme ne s'attaque pas aux mêmes problématiques que celles citées au dessus.

Jack Intelligent Agents d'Agent Oriented Software[2] :

Jack Intelligent Agents est destiné au développement d'agent de type BDI (Belief, Desire, Intention), tout en fournissant une API orientée objets. Un agent BDI (Beliefs, Desires and Intentions) est caractérisé par ses croyances qui correspondent aux connaissances qu'il a du monde, par ses désirs, à savoir les objectifs accessibles en fonction de ses croyances et de ses intentions, et finalement par ses intentions, qui caractérisent les objectifs courants ou en cours de réalisation de

[1] http://www.swarm.org
[2] http://www.agent-software.com.au

l'agent. Pour cela, un langage proche de la programmation logique est proposé au développeur, qui peut ensuite le compiler pour le transformer en classes Java. Les classes générées étendent des classes fournies par l'Api du noyau de Jack, à savoir, la classe Agent, la classe Event ou encore la classe Plan. Le noyau de Jack gère aussi la concurrence des tâches entre les agents, la réaction aux événements et l'infrastructure de communication.

Jade du Tilab[3] :

Jade, pour Java Agent DEvelopment framework, est un middleware écrit en Java et se conformant aux spécifications de la FIPA (Foundation for Intelligent Physical Agents, voir annexe A). Cet environnement simplifie le développement d'agent en fournissant les services de base définis par la FIPA, ainsi qu'un ensemble d'outils pour le déverminage et le déploiement.

Une plateforme Jade peut être répartie sur un ensemble de machines et configurée à distance. La configuration du système peut être modifiée dynamiquement, grâce à un mécanisme de migration d'agent au sein de la même plateforme.

Voyager de Reticular Systems[4] :

L'environnement Voyager peut être considéré comme un ORB, Object Request Broker, orienté agents mobiles. En plus de la gestion de l'invocation à distance des méthodes d'un agent mobile, Voyager propose des mécanismes de persistance des messages, de gestion de la sécurité ou encore la gestion des communications entre agents utilisant différents protocoles (RMI et IIOP par exemple).

AgentBuilder de Reticular Systems[5] :

La plateforme AgentBuilder est une implémentation assez proche du langage 'Agent-0' proposé par [SHO 93] . Les agents sont décrits avec le langage Radl, Reticular Agent Definition Language, qui permet de définir les règles du comportement de l'agent. Les règles se déclenchent en fonction de certaines conditions et sont associées à des actions.Les conditions portent sur les messages reçus par l'agent tandis que les actions correspondent à l'invocation de méthodes Java. Il est aussi possible de décrire des protocoles définissant les messages acceptés et émis par l'agent. AgentBuilder est probablement la plateforme commerciale la plus aboutie et la plus rentable.

[3] http://jade.tilab.com/
[4] http://www.recursionsw.com/voyager.htm
[5] http://www.agentbuilder.com

MadKit du Lirmm[6] :

La plateforme MadKit développée à l'Université de Montpellier II, est basée sur la notion d'agent, de groupe et de rôle. MadKit fournit une Api permettant la construction d'agent en spécialisant une classe d'agent abstraite. Chaque agent peut tenir différents rôles au sein de différents groupes. Les agents sont lancés par le noyau de MadKit, qui propose notamment les services de gestion des groupes et de communication. Il est ainsi possible d'échanger des messages directement à un agent ou à l'ensemble d'un groupe. Cette plateforme est surtout intéressante pour l'approche organisationnelle qu'elle met en avant lors de l'analyse et de la conception d'un système multi-agents.

Volcano de Magma :

La plateforme Volcano développée par Pierre-Michel Ricordel dans le cadre de sa thèse, est basée sur la méthodologie Voyelles [DEM 01] et une approche componentielle. La méthodologie Voyelles repose sur quatre concepts, identifiés par les quatre voyelles : A pour Agents, E pour Environnements, I pour Interactions et O pour Organisations.

Un langage de description d'architecture de composants (Madel) est défini pour déclarer les dépendances entre les différentes briques constituant le comportement de l'agent. Cette plateforme est intéressante car les problématiques de génie logiciel liées à la réutilisabilité sont abordés et favorisés par l'utilisation de composants logiciels.

Zeus de British Telecom[7] :

Zeus fournit un environnement de développement d'agent grâce à un ensemble de bibliothèques Java que les développeurs peuvent réutiliser pour créer leurs agents. Zeus propose aussi un ensemble d'agents utilitaires (serveur de nommage et facilitateur) pour faciliter la recherche d'agents.

L'architecture de Zeus se prête bien aux applications de planification ou d'ordonnancement, ce qui la rend particulièrement intéressante pour la réalisation de simulation ou d'applications de supervisation. Cependant, chaque agent fonctionne grâce à une machine virtuelle Java (Jvm, Java Virtual Machine), ce qui devient rapidement difficile à gérer lorsque le nombre d'agents dans le système augmente.

[6] http://www.madkit.org
[7] http://www.labs.bt.com/projects/agents/zeus/

Dans le paragraphe suivant, nous commençons par présenter quelques définitions relatives au TCAO. Ensuite, nous passerons en revue les modèles de conception et d'architecture pour ce type de systèmes.

1.3 Travail collaboratif Assisté par Ordinateur : TCAO

Pour assister la collaboration, les recherches sur le travail collaboratif assisté par ordinateur (TCAO) proposent des outils de communication, de travail et d'échange ou de partage de données permettant de réaliser un fort couplage entre participants et ainsi de créer une véritable intelligence collective dans les organisations.

Dans ce qui suit nous présenterons quelques définitions nécessaire à la compréhension de cette thèse.

1.3.1 Collaboration

Dans le domaine de l'enseignement/apprentissage, le travail collaboratif entre apprenants et/ou enseignants se concrétise le plus souvent par un travail d'équipe, "*l'équipe étant perçue comme étant un groupe de personnes interagissant afin de se donner ou d'accomplir une cible commune, laquelle implique une répartition de tâches et la convergence des efforts des membres de l'équipe*" [ALA 96]. Dans le cas où la cible commune d'un travail d'équipe est un but ultime à atteindre nous parlerons de travail coopératif visant l'apprentissage, lequel peut se définir comme suit : "*l'apprentissage coopératif est une activité d'apprentissage en groupe, organisée de façon à ce que l'apprentissage soit dépendant de l'échange d'informations socialement structuré qui s'effectue entre les apprenants du groupe. C'est également une activité dans laquelle l'apprenant est responsable de son propre apprentissage et motivé pour participer à l'apprentissage des autres.*" [LOP 99]. Les tâches coopératives en formation "*supposent l'assignation d'une tâche collective exercée en groupe restreint, exigeant un maximum d'interactions entre pairs, sans la supervision directe et immédiate du formateur*" [CAR 99]. Ainsi, contrairement à une collaboration, une coopération n'engendre pas nécessairement la création d'une œuvre commune. Nous parlerons de travail collaboratif lorsque la cible commune du travail d'une équipe consiste, outre le travail en groupe, en la réalisation d'un produit final. Par travail collaboratif, nous désignons donc, d'une part, la coopération entre les membres d'une équipe et, d'autre part, la réalisation d'un produit fini. La coopération, entre les membres d'une équipe, est déterminée

par une communication et une coordination entre tous les membres de l'équipe, et ce sans centralisation, ni de l'une ni de l'autre. La figure 1.1 illustre bien cette définition de la collaboration.

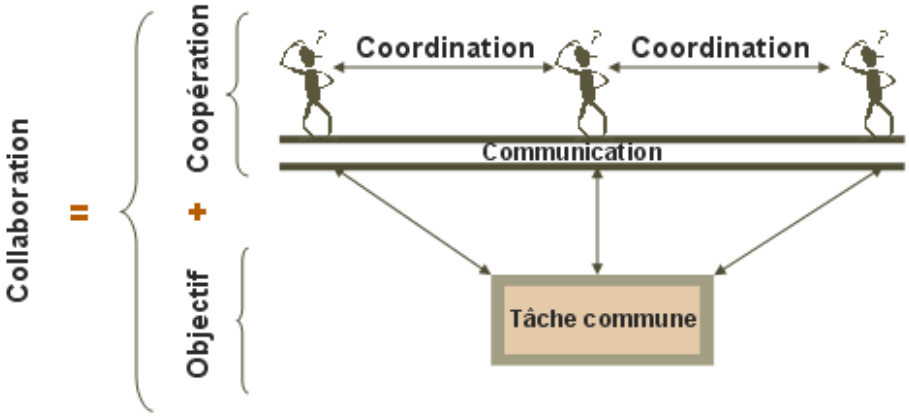

Figure 1.1: Illustration du principe de la collaboration

1.3.2 Les modèles de conception des collecticiels

Bien que plusieurs personnes se soient attachées à tenter de définir précisément le terme *collecticiel*, celui-ci n'a toujours pas reçu de définition précise et consensuelle. Selon Ellis, *"un collecticiel est un système informatique qui assiste un groupe de personnes engagées dans une tâche commune et qui fournit une interface à un environnement partagé"* (traduction de A. Karsenty [KAR 94]). Pour David Colleman, il s'agit d'*"un outil qui permet la collaboration à l'aide de l'ordinateur en vue d'augmenter la productivité ou la fonctionnalité des processus personne à personne"* [COL 92]. La grande différence entre ces deux définitions est que la première fait intervenir explicitement l'interface utilisateur comme une des composantes essentielles d'un collecticiel. Cette interface décrit à la fois l'interaction entre l'homme et le système et l'interaction homme-homme au travers de ce système ainsi qu'il est précisé dans [ELL 94]. Cette première définition est également indépendante des moyens mis en œuvre pour assister les individus, en particulier elle n'indique pas les fonctionnalités à couvrir avec un collecticiel. Dans le but d'identifier au moins quelques unes de ces fonctionnalités, les collecticiels doivent non seulement assister un groupe de personnes, mais ils doivent également en améliorer l'efficacité [SIR 00].

Certaines recherches ont consisté à analyser les caractères spécifiques des composants d'un travail ou d'un phénomène collaboratif. Les résultats de cette analyse ont permis de proposer les définitions suivantes :

Le groupe et l'individu : Chaque participant est un individu personnalisé qui appartient à un ou plusieurs groupes de participants. Un groupe est un ensemble d'individus travaillant sur un même domaine. Suivant le degré de cohésion du groupe, Bellamine [BEL 96] préfère utiliser les termes de collectif, groupe ou équipe.

Les rôles : Dans chaque groupe et à un instant donné, chaque participant joue un rôle [DAV 92]. Ce rôle est caractérisé par l'ensemble des droits et des devoirs du participant vis-à-vis de ses partenaires et des données partagées. Il peut évoluer au cours du temps et être constitué de plusieurs rôles élémentaires.

Les vues : Chaque participant peut avoir sa propre perception (vue) des entités manipulées collectivement. La notion de vue inclut à la fois des choix de représentation, et des choix d'interaction [MAR 94]. En fait, une vue peut être considérée comme un filtre bidirectionnel sur un ensemble d'entités partagées et elle définit la représentation des entités vers l'utilisateur et autorise ou interdit les accès à ces entités. Une vue peut être publique (accessible par d'autres utilisateurs), privée (accessible seulement par le propriétaire) ou semi-privée (publique pour un sous-groupe ou intégrant quelques éléments publics et d'autres privés, etc.).

Le WYSIWIS (What You See Is What I See) : Il a pour but d'assurer la rétroaction du groupe, c'est-à-dire de permettre à chaque participant de voir ce qu'un autre participant voit ou fait. Cette notion est en fait modulable. Lorsque la vision est vraiment identique chez plusieurs participants, on parle de WYSIWIS strict, tandis que lorsque la vision d'une même situation est différente, on parle de WYSIWIS relâché.

1.3.2.1 Le trèfle fonctionnel

Le modèle du tréfle fournit un cadre conceptuel utile pour déterminer les requis fonctionnels et mener une analyse fonctionnelle. Au début, les chercheurs ont proposé deux ensembles fonctionnels distincts, l'espace de production et l'espace de coordination.

L'espace de production désigne les objets résultant d'une activité de groupe

(un document par exemple). Pour Ellis [ELL 91], cet espace de production concerne le résultat des tâches communes à accomplir et pour Coleman [COL 92], cet espace est celui dans lequel la productivité doit avoir lieu.

Par contre l'espace de coordination est celui qui définit les acteurs et leur structure sociale, ainsi que les différentes tâches à accomplir en vue de produire les objets de l'espace de production.

Par la suite Ellis [ELL 94] a complété ce modèle par un troisième espace, celui de la communication. Il offre aux acteurs de l'espace de coordination la possibilité d'échanger de l'information dont la sémantique concerne exclusivement les acteurs, le système n'étant qu'un messager.

Ce modèle de classification est appelé trèfle fonctionnel, selon ce modèle, un collecticiel couvre trois espaces fonctionnels, l'espace de production, de communication et de coordination (figure 1.2).

Figure 1.2: Le modèle du trèfle fonctionnel

L'espace de communication : Cet espace correspond aux fonctionnalités permettant l'échange d'information entre les acteurs du collecticiel. Cet échange est de la communication homme-homme médiatisée [SAL 95]. Tarpin dans [TAR 97] a limité le mot communication à l'échange de données informatiques, et il a parlé de conversation lorsqu'il s'agit d'un échange interpersonnel de signaux, numériques ou analogiques, ne pouvant être interprétés par la machine. Cependant il existe différents modes de communication comme, par exemple, l'audio (téléphone), la vidéo (mediaspace), le textuel (messageries), la gestuelle (langue des signes) ou l'haptique (communication à travers un système à retour d'effort comme, par exemple, le système InTouch [BRA 98]).

L'espace de coordination : Cet espace correspond aux fonctionnalités dédiées à l'assignation de tâches et de rôles aux différents acteurs d'une activité collaborative. Ces fonctionnalités ont pour but de coordonner les acteurs afin de réaliser

une œuvre commune. Cette coordination peut s'exprimer en terme de planification de tâches.

L'espace de production : Cet espace concerne l'ensemble des fonctionnalités de production d'objets partagés tels que des documents communs et la gestion des accès à ces données partagées. Par exemple, les éditeurs partagés, sont dédiés à la production.

En d'autres termes, ces trois espaces correspondent aux services nécessaires pour communiquer, coordonner et produire. En effet, l'espace de communication correspond aux services nécessaires pour recréer *l'espace des personnes*. Ce dernier contient toutes les informations qui servent aux communications interpersonnelles comme l'image des participants ou l'espace auditif, etc. L'espace de coordination a été ajouté pour tenir compte des services logiciels liés à la *gestion des mécanismes de coordination*. Alors que l'espace de production correspond aux services nécessaires pour recréer *l'espace de la tâche* [BUX 92]. L'espace de la tâche est centré sur les outils, les documents, et les artefacts produits en commun.

Le modèle du trèfle est un modèle fonctionnel du système, c'est-à-dire qu'il décrit les classes de fonctions qu'un collecticiel doit implémenter au niveau logiciel. Il peut servir à décrire les liens entre les fonctionnalités d'une application et l'architecture logicielle sous-jacente [CAL 97]. Même si le modèle du trèfle ne devait s'appliquer qu'à une description fonctionnelle du logiciel, et donc intervenir après la description de l'activité collaborative, en pratique il oriente l'analyse du travail coopératif. En effet, pour que les services logiciels soient adaptés aux besoins réels de la collaboration, ils doivent se reconnaître dans une description des activités collaboratives. Pour appliquer le modèle du trèfle fonctionnel afin de concevoir des applications, il faut être en mesure de transformer la description des activités d'un groupe donné suivant les trois facettes communication, production et coordination.

1.3.2.2 Evolution dans le temps de l'importance des différents espaces

Les trois espaces fonctionnels d'un collecticiel étant définis, il convient néanmoins de noter que certaines fonctionnalités peuvent être à l'intersection de plusieurs espaces comme le souligne la figure 1.2. Ainsi, la distinction selon les trois espaces n'est pas stricte car il est tout à fait possible qu'une activité de coordination ait lieu suite à une activité de communication. Par exemple, la prise de rendez-vous par téléphone est une activité de coordination basée sur une activité de communication.

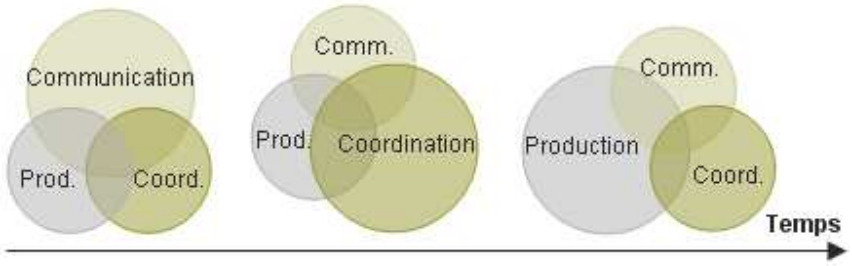

Figure 1.3: Evolution possible dans le temps de l'importance des différents espaces.

De plus, l'importance des trois espaces d'un collecticiel, comme le montre la figure 1.3, peut évoluer au cours du temps [LAU 02] : cette évolution peut être représentée grâce au modèle du trèfle. Par exemple, lors de l'utilisation d'un système dédié à la production, il est possible, à un moment donné, que l'activité de groupe soit centrée sur la communication en vue de se coordonner afin de redéfinir l'activité de production. Cette approche permet de prendre en compte la variabilité dans les activités de groupe au cours du temps.

Comme nous l'avons évoqué, le modèle du trèfle est un modèle de conception utile pour déterminer et organiser les fonctionnalités d'un collecticiel selon la production, la communication et la coordination. Par exemple, F. Tarpin propose une méthode pour spécifier les fonctionnalités d'un collecticiel en fonction d'un ensemble de questions thématiques, en s'appuyant sur les trois espaces fonctionnels du modèle du trèfle [TAR 97] :

- Production : structuration et gestion des données, par identification des données partagées et de leurs structures, par modélisation du travail et des opérations de production et par identification des caractéristiques de partage.
- Coordination : organisation du processus de travail par identification des tâches et modélisation du processus de travail et par identification des rôles et des modes de coopération (couplage).
- Communication : choix des modes de communication et des protocoles de conversation (rôles et attribution de responsabilités).

1.3.3 Classification des collecticiels

Une façon de classer les collecticiels a été d'établir une typologie des groupes. Cette approche a donné la classification espace-temps des collecticiels tout en créant des outils adaptés à chaque groupe.

1.3.3.1 Classification Espace/Temps

L'approche traditionnelle pour présenter les collecticiels consiste à utiliser la classification espace-temps d'Ellis [ELL 91]. Suivant cette classification, les systèmes se différencient par la distance temporelle entre les utilisateurs et par leur distance spatiale. Les utilisateurs interagissent de manière synchrone s'ils manipulent simultanément les données partagées, ou de manière asynchrone s'ils les manipulent à des moments différents. De même ils peuvent interagir en étant dans la même pièce ou dans des lieux différents.

La matrice "espace-temps" du tableau 1.1 contient deux axes : le premier axe "espace" considère la distance spatiale entre les utilisateurs "Lieu identique-Lieux différents" et le deuxième axe "temps" considère la distance temporelle entre les utilisateurs "Temps identique - Temps différents" (ou encore "Synchrone - Asynchrone"). Cependant, le découpage spatio-temporel n'est pas toujours aussi net. Dans [DOU 92], un cas de coordination intermédiaire est souligné. Donner des informations sur les activités présentes d'utilisateurs en ligne et en même temps des informations sur les activités passées des utilisateurs déconnectés conduit à voir sous deux angles différents la coordination. D'une part elle est synchrone entre les utilisateurs en ligne, d'autre part elle est asynchrone entre les utilisateurs en ligne et ceux qui n'y sont plus. Cette coordination intermédiaire est qualifiée de semi-synchrone. De plus, pour tenir compte de l'aspect variable et flexible du découpage spatio-temporel, des valeurs indéterminées ont été introduites sur les deux axes dans [GRU 94]. Il s'agit de lieux différents et imprévisibles et du temps d'utilisation différent et imprévisible. L'aspect imprévisible se résume à une mise en correspondance entre l'interaction et les attentes et suppositions des utilisateurs. Un aspect est imprévisible si les utilisateurs ne peuvent prévoir à priori la valeur de cet aspect. Par exemple, lors d'une collaboration sur l'écriture d'un document sans date butoir, il n'est pas possible de prévoir quand le document va changer d'éditeur. C'est un cas de temps d'utilisation différent et imprévisible. L'aspect prévisible et imprévisible est lié aux contraintes imposées aux utilisateurs. Ceci fait que le temps d'utilisation à travers un courriel est indéterminé, car l'expéditeur ne peut pas savoir quand le destinataire lira le message. A l'inverse, dans un workflow avec des limites de temps imposées, les temps d'utilisation sont prévisibles. Néanmoins, comme il est souligné dans [GRU 94], il convient de ne pas s'enfermer dans une catégorie, car les activités passent souvent d'une catégorie à une autre. Cette souplesse peut se révéler décisive dans l'utilisabilité des collecticiels.

L'ampleur du World Wide Web illustre parfaitement l'évolution et l'importance grandissante des nouvelles technologies communicantes. Initiés par l'essor des

		T E M P S		
		Temps identique	**Temps différents et prévisibles**	**Temps différents et imprévisibles**
E	**Lieu identique**	Type (1)	Type (2)	Type (3)
S				
P	**Lieux différents et prévisibles**	Type (4)	Type (5)	Type (6)
A				
C	**Lieux différents et imprévisibles**	Type (7)	Type (8)	Type (9)
E				

Tableau 1.1: Découpage Espace Temps

technologies communicantes, de nombreux collecticiels sont conçus dans des domaines variés comme l'enseignement (connu sous le nom anglais de CSCL, Computer-Supported Cooperative Learning) ou les jeux (connu sous le nom anglais de CSCP, Computer-Supported Cooperative Play). Ainsi, grâce à cette capacité grandissante des moyens de communication, les études peuvent se concentrer sur les systèmes collaboratifs asynchrones et/ou synchrones qui font apparaître de nombreux enjeux sociaux et économiques. La notion de groupe est alors la clé de voûte de ces collecticiels et les philosophes parlent de conscience collective.

La matrice résultante considère trois cas pour chaque axe. Nous illustrons dans ce qui suit les neuf types de collecticiels obtenus en fonction du type d'application.

1.3.3.2 Quelques applications

Communication homme-homme médiatisée "CHHM" :

Les messageries électroniques sont actuellement les collecticiels les plus répandus et les plus utilisés et dans leur forme usuelle elles sont de type (6). Il est comparable au courrier postal, les destinataires sont répartis géographiquement, mais sont localisables. Ainsi, lorsque l'on envoie un message électronique, l'envoyeur peut trouver le destinataire sans connaître sa localisation. Par contre, le moment de réception est imprévisible.

D'autres formes de courrier électronique sont du type (9). Par exemple, les listes de diffusion et les spam ne permettent pas la localisation des participants. La conséquence en est que les outils de messagerie se sont enrichis de fonctionnalités "intelligentes" pour trier les courriers, détruire les courriers non désirables

ou pour envoyer des réponses automatiquement.

Les systèmes de vidéoconférence sont des collecticiels de type (4). L'interaction entre les utilisateurs a lieu en temps réel (même temps) et les participants sont répartis géographiquement dans des lieux différents mais prévisibles. Il s'agit d'un forum de discussion offrant une communication reposant sur des données audio et vidéo. La grande difficulté du déploiement de ce type d'application est liée en grande partie à la nécessité de disposer d'une bande passante capable de diffuser et recevoir des données audio et vidéo avec une qualité acceptable. Le système PictureTel[8] est un des systèmes le plus connu.

Les mediaspace [COU 99] [MAC 99] sont des collecticiels du type (4). Ils mettent en œuvre une liaison vidéo au sein d'une équipe dans le but de favoriser la communication informelle et d'entretenir une conscience de groupe forte entre membres. L'objectif visé est différent des systèmes de vidéoconférence bien que les deux types d'application reposent sur des flux vidéo. En effet, contrairement à la vidéoconférence qui met en relation des individus sur une courte période et de manière planifiée, la connexion vidéo d'un mediaspace est permanente et l'interaction est opportuniste.

Le post-it est un collecticiel de type (3), qu'il soit réel (papier) ou virtuel, il reste un support de la collaboration très largement adopté. La transmission du post-it est totalement imprévisible et se déroule, au sens large, dans un même lieu. Par extension, nous pouvons supposer que le support d'un post-it, comme un livre, est toujours le lieu de l'interaction bien qu'il soit possible de le déplacer.

Applications aux jeux :

Les jeux sont certainement les collecticiels qui connaissent, avec les systèmes de messagerie, l'essor le plus fulgurant. Les consoles de jeux ou les bornes d'arcades multi-joueurs sont des formes de collecticiel de type (1) c.a.d synchrone en face-à-face. Ces consoles sont constituées de deux manettes (ou plus) et chaque joueur pilote un personnage de jeu ou un objet animé. Il existe également des jeux de type (7), ce sont généralement des jeux en réseau où l'emplacement des joueurs est imprévisible. Ces jeux misent sur la coopération et la compétition entre les joueurs. Ce type d'application est comparable à une forme d'éditeur partagé reposant, dans la majorité des cas, sur un mode de communication textuelle.

Applications pour la coordination :

Les systèmes workflow [FLO 88], [BOW 95], [ELL 99] sont du type (5). Ils ont pour objectif la coordination des activités et des intervenants au cours d'un

[8] http://www.picturetel.com

processus (industriel, administratif, etc). Ces systèmes permettent ainsi la planification des interventions de chacun sur les documents échangés au cours d'un processus. Dans cette situation, les participants sont répartis géographiquement mais localisables (le bureau sur le lieu de travail). Dans cette même catégorie, nous pouvons citer également les systèmes d'aide à la décision (GDSS, Group Decision Support Systems) qui peuvent être de type (2) et les calendriers partagés (group calendars) [PAL 02]. Les premiers sont conçus pour faciliter la prise de décisions et les seconds offrent des services de planification de tâches et de gestion de projets.

Applications d'édition :

Les éditeurs partagés sont des systèmes de type (8) car il n'est pas nécessaire de connaître la localisation exacte des différents auteurs du document. Par contre, pour pouvoir transmettre le document entre les auteurs, ces derniers doivent planifier leurs échanges. Ces outils sont complexes à réaliser, en particulier pour la gestion des tâches concurrentes comme le "défaire" et "refaire" ou la fusion de différentes versions [DOU 96]. Les éditeurs de texte partagés comme le système StorySpace[9] ou les éditeurs de dessins partagés comme les tableaux blancs partagés [ISH 94] (shared whiteboard) sont d'autres exemples de collecticiels permettant l'édition conjointe.

1.3.4 Les modèles d'architecture pour les collecticiels

Le terme architecture logicielle décrit généralement, la structure des composants d'un programme / système, leurs relations, et les principes et lignes directrices caractérisant leur conception et leur évolution au cours du temps. Malgré une évolution continue et un progrès récent considérable, les architectures des collecticiels utilisées actuellement sont pour la plupart basées sur des modèles anciens et elles dérivent de trois grandes familles largement décrites dans la littérature à savoir les modèles monolithiques, les modèles multi-agents et les modèles hybrides. Nous allons maintenant présenter rapidement les différentes architectures pour chaque type de modèle. Nous commençons par introduire les modèles des systèmes interactifs, à partir des quels, les modèles de collecticiel sont dérivés. Ensuite, nous exposons les modèles d'architecture des collecticiels.

[9] http://www.eastgate.com

1.3.4.1 Les modèles d'architecture pour les systèmes interactifs

Un modèle monolithique : Le modèle de référence et qui est à l'origine de nombreux modèles d'architecture pour les collecticiels est le modèle des systèmes interactifs -Arch- [BAS 92], qui est lui même une extension du modèle séminal Seeheim [GRE 85] . Le principe de ce modèle est de séparer l'interface utilisateur du Noyau Fonctionnel et propose ainsi une décomposition canonique des principaux composants d'un système interactif. En effet, le modèle Arch structure un système interactif selon cinq niveaux d'abstraction, distinguant ceux qui relèvent du domaine de l'application de ceux qui gèrent l'interface utilisateur.

Un modèle multi-agent : D'autres modèles d'architecture de systèmes interactifs, dits modèles multi-agents, préconisent une décomposition fonctionnelle plus fine. Le modèle multi-agent MVC a été introduit dans le langage Smalltalk [KRA 88], il est basé sur le même principe que le modèle Arch car il distingue la partie interface utilisateur du modèle de l'application (Noyau Fonctionnel). Dans ce modèle un agent est constitué de trois facettes. Une facette modèle (M) qui représente les concepts du domaine. Une deuxième facette contrôleur (C) qui interprète les entrées au niveau de l'interface utilisateur. La troisième est une facette vue (V) qui offre une représentation en sortie au niveau de l'interface utilisateur (affichage, son, haptique, etc).

Un modèle hybride : Parallèlement aux deux types d'architectures exposés dans les paragraphes précédents une autre c'est développé en combinant les propriétés de chacun d'eux. Les modèles hybrides sont généralement une extension de modèle Arch selon une approche multi-agent inspiré du modèle PAC [COU 94]. Nigay a proposé un système interactif basé sur ce type de modèle appelé PAC-Amodeus [NIG 94]. Ce modèle reprend les cinq niveaux fonctionnels du mo-dèle Arch et structure le composant qui a la charge de gérer le dialogue avec une hiérarchie d'agent PAC. Rappelant qu'un agent PAC est composé de trois facettes, Abstraction (A), Présentation (P) et Contrôle (C). La facette A gère les concepts du domaine est définit la compétence de l'agent. La facette P définit l'interface utilisateur et interprète les entrées/sorties générées par l'utilisateur et la facette C sert de lien et de canal de communication entre les deux autres.

1.3.4.2 Les modèles monolithiques pour les collecticiels

Patterson [PAT 94] a proposé le modèle Zipper pour décomposer les collecticiels. Ce modèle repose sur la notion d'états partagés, il définit quatre niveaux d'abstraction et chacun d'eux représente un état. Le premier est l'état de l'écran

(Display), il correspond à l'état des périphériques d'entrée et de sortie (moniteur, souris, etc). Le second est l'état de la vue (View) qui correspond à l'état de la présentation logique des données, c'est-à-dire l'état de l'interface utilisateur. Le troisième est l'état du modèle (Model) qui correspond au Noyau Fonctionnel et aux objets du domaine, et en dernier, nous trouvons l'état du fichier (File) qui correspond à la représentation persistante du modèle.

Une autre façon de décomposer les collecticiels est le méta-modèle de Dewan [DEW 99] qui est au fait une généralisation des deux modèles Arch et Zipper. Selon ce méta-modèle, un collecticiel est constitué d'un nombre variable de couches représentant plusieurs niveaux d'abstraction. Dewan propose un moyen pour identifier à quel niveau un collecticiel doit mettre en œuvre la collaboration, à savoir si la collaboration relève du Noyau Fonctionnel ou si elle relève plus de la Présentation. Un degré de collaboration (collaboration awareness degree) permet ainsi de déterminer les couches susceptibles de générer des actions concurrentes et de déterminer à quel niveau il est nécessaire d'implémenter des mécanismes de contrôle d'accès, de verrouillage des données ou de couplage des données.

1.3.4.3 Les modèles multi-agents pour les collecticiels

A partir de du modèle MVC, est développé un modèle de collecticiel multi-agent basée sur une approche par composant nommé Clock [GRA 97]. Comme pour un agent MVC, un composant Clock est composé de trois facettes. Une Facette Modèle gère les objets du domaine encodés sous forme de données de type abstrait (ADT, Abstract Data type) et les deux facettes contrôleur et Vue servent à interpréter les interactions de l'utilisateur avec le système et à gérer le rendu en sortie. La différence entre un composant Clock et un agent MVC réside sur le fait que les deux facettes contrôleur et Vue ne communiquent pas entre elles. De plus, la communication de ces deux facettes avec le modèle est désormais monodirectionnelle. Le contrôleur communique au modèle toutes les interactions faites par l'utilisateur et la vue reçoit les modifications de mise à jour de la présentation.

Le Modèle Arch a également inspiré le modèle multi-agent ALV à partir duquel est réalisé la plateforme Rendez-Vous [HIL 94]. Cette dernière est une infrastructure permettant le développement d'applications collaboratives reposant sur la notion d'objets partagés. Le collecticiel ainsi développé, selon la terminologie du modèle Arch, serait composé d'un unique Noyau Fonctionnel partagé pour plusieurs instances de la présentation. Dans ce modèle, les agents sont composés des trois facettes Abstraction partagé, Vue et Lien. La première gère les objets du domaine partagés par tous les utilisateurs, la seconde interprète les entrées d'un utilisateur et gère les sorties. La dernière facette, comme son nom

l'indique, sert à relier les deux premières et aussi de s'assurer de la conformité de leurs données. Dans cette même catégorie, nous pouvons également citer le modèle AMF-Collaboratif [TAR 97] qui est une extension du modèle AMF Agent Multi-Facettes [OUA 94] reposant sur une approche multi-agent mutlifacettes.

1.3.4.4 Les modèles hybrides pour les collecticiels

A partir du modèle PAC-Amodeus, une extension pour les collecticiels a été proposée par Salber [SAL 95]. Dans ce modèle nommé CoPAC, deux types d'agents subsistent, l'agent PAC usuel mono-utilisateur et l'agent collaboratif communiquant avec les autres agents collaboratifs. Il s'agit d'un agent PAC augmenté d'une nouvelle facette communication permettant aux agents collaboratifs de communiquer entre eux directement. La facette Contrôle se charge alors de distribuer les messages en provenance des facettes Abstraction et Présentation vers la facette Communication et réciproquement. L'ajout de cette nouvelle facette s'accompagne de l'introduction d'un niveau d'abstraction supplémentaire au modèle Arch avec l'existence d'un composant communication.

Dans [CAL 97] Calvary et al ont proposé une décomposition plus affinée d'un agent PAC selon les trois espaces fonctionnels du modèle du trèfle (production, communication et coordination). Dans ce modèle appelé PAC*, les trois facettes d'un agent PAC sont alors décomposées en trois parties dédiées chacune à une dimension du modèle du trèfle. Par conséquent, un agent couvre un espace fonctionnel selon deux dimensions orthogonales, les trois facettes de l'agent PAC et les trois dimensions du modèle du trèfle. Selon cette décomposition, un agent PAC* peut donc exister sous trois formes différentes : la forme centralisée, la forme répartie et la forme hybride. Dans la forme centralisée, un agent PAC* est composé de trois agents dédiés respectivement à la production, à la communication et à la coordination et reliés à un agent ciment. Ce dernier assure la communication entre les trois agents et le reste du monde, c'est-à-dire les autres agents de la hiérarchie. Dans la forme hybride, les trois agents dédiés communiquent directement entre eux et assurent eux-même la liaison avec les autres agents de la hiérarchie. Enfin, la forme hybride est la combinaison des deux formes précédentes, sachant que l'agent ciment conserve toujours son rôle de lien avec le monde extérieur.

1.3.4.5 Classification des modèles d'architecture pour les collecticiels

Une architecture logicielle ne peut pas être bonne ou mauvaise; cependant sa qualité se juge suivant les attentes de l'utilisateur final du logiciel et des qualités de ce dernier définies par le concepteur.

Nous allons comparer les architectures déjà présentées suivant nos attentes pour notre système de télétravail par Internet. Pour cela, nous énumérons les qualités requises pour comparer les différentes architectures (inspirées de la grille d'analyse de Laurillau dans [LAU 02]). Une architecture conforme aux attentes du travail de groupe doit mettre en évidence les éléments suivants :

- Faire la différence entre les actions individuelles et les actions collectives à savoir, classer les actions collectives selon l'un ou l'ensemble des trois espaces fonctionnels (production, communication et coordination).
- Les types des composants qui gèrent les données, c'est-à-dire, les composants privés pour les données (ressources) privées et les composants publics pour les données collectives.
- L'observabilité des actions et des données, c'est-à-dire, si un utilisateur peut voir ce que fait tous les participants ou pas et si oui comment il peut le voir.

Dans le tableau 1.2, nous avons classé les différentes architectures logicielles des collecticiels suivant les critères cités au dessus. Dans ce tableau, nous remarquons que les différentes architectures sont plus au moins compatibles avec les attentes du travail de groupe. Les modèles DEWAN, CLOVER, COPAC et PAC* font la différence entre les actions individuelles et les actions collectives. Tandis que les deux premiers prennent en compte les actions individuelles, les deux autres ne les représentent pas. Seulement deux modèles (PAC* et CLOVER) représentent les actions collectives selon les trois espaces du trèfle fonctionnel. Tous les modèles font la différence entre les ressources collectives et les ressources individuelles. Presque tous les modèles présentés, montrent, aux différents utilisateurs, les actions passées des uns aux autres. Trois modèles permettent de rendre l'utilisation des données (ressources) publiques (le Méta-modèle de DEWAN, le modèle PAC* et le modèle CLOVER).

Nous venons de présenter les architectures logicielles des collecticiels et nous constatons l'existence d'une analogie certaine entre les environnements de télétravail collaboratif et les systèmes multi-agents (SMA). L'un des principaux objectifs du TCAO est de permettre l'interaction entre de multiples participants immergés dans des mondes virtuels peuplés d'entités virtuels. Par conséquent, les sujets participant à la collaboration peuvent être comparés à des agents, d'autant plus que le monde virtuel est très proche du concept d'environnement dans les SMA.

1.4 Bilan et conclusion de l'état de l'art

Ce premier chapitre a comme objectif principal la recherche des concepts, des méthodes et des outils nécessaires pour répondre à la problématique suivante :

Les modèles d'architecture	Actions individuelles	Actions collectives	Observabilité des actions	Observabilité des ressources
ZIPPER	Pas de différence		Oui	Non
DEWAN	Oui	Oui	Oui	Oui
ALV	Pas de différence		Oui	Non
CLOCK	Pas de différence		Oui	Non
AMF-C	Pas de différence		Oui	Non
COPAC	Non	Oui (3 niveaux)	Oui	Non
PAC*	Non	Oui (3 niveaux)	Oui	Oui
CLOVER	Oui	Oui	Oui	Oui

Tableau 1.2: Tableau comparatif des différents modèles

Comment concevoir un système capable de prendre en charge la collaboration de plusieurs utilisateurs distants pour préparer et réaliser des missions de téléopération ?

Le système doit :

- Supporter un travail collaboratif.
- Assister et superviser les utilisateurs pendant le déroulement de la mission.
- Permettre l'analyse et l'évaluation de la collaboration au sein d'un groupe et/ou de l'ensemble des groupes.
- Permettre la réalisation des tâches d'une manière synchrone (temps réel) et/ou asynchrone (temps différé) en fonction de la mission.
- Posséder une architecture logicielle ouverte pouvant être utilisée dans d'autres domaines d'applications.

Afin de tenter de chercher des éléments indispensables pour bien mener cette problématique, nous avons abordé deux domaines de recherche qui sont les Systèmes Multi-Agents (SMA) et le Travail Collaboratif Assisté par Ordinateur (TCAO).
Les études menées dans le domaine des SMA ont deux objectifs. Le premier consiste à présenter les différents concepts, propriétés et formalismes des SMA utiles pour la modélisation des systèmes complexes et distribués. Le second objectif vise à établir un bilan des plateformes de développement multi-agents pouvant être utilisées pour l'implémentation de tels systèmes.
Dans le domaine du TCAO, l'étude présentée a comme objectifs de tenter de répondre aux questions suivantes :

- Quelle définition peut-on attribuer au terme "collaboration" ?
- Comment peut-on modéliser la collaboration ?

- Quels sont les modèles d'architectures existants ?

Cet état de l'art nous a permis d'identifier les différents éléments qui serviront à la réalisation de notre système de téléopération collaborative tout en respectant les contraintes imposées par notre problématique. Ces différents éléments sont récapitulés ci-dessous :

1) Le formalisme de Ferber pour la spécification formelle de notre système de collaboration.
2) Le trèfle fonctionnel des collecticiels pour la spécification fonctionnelle de la collaboration.
3) Le principe du modèle d'architecture PAC* pour la modélisation de notre architecture de collaboration.
4) La plateforme de développement multi-agent JADE pour l'implémentation de notre SMA pour la Collaboration (SMA-C).

Dans le chapitre suivant, nous présentons le système multi-agent pour la collaboration que nous appelons SMA pour la Collaboration (SMA-C). Nous détaillerons à cette occasion les éléments 1) et 2) qui sont utilisés pour la modélisation et dans le chapitre 3 les éléments 3) et 4) utilisés respectivement pour la conception et l'implémentation du SMA-C.

Chapitre 2

Modélisation d'un Système Multi-Agent pour la Collaboration (SMA-C)

2.1 Introduction

Dans le chapitre précédent, nous avons montré la nécessité de concevoir un système dédié à la collaboration. Un système multi-agent qui prend en compte les caractéristiques de la collaboration et qui permet le travail collaboratif synchrone de plusieurs personnes distantes. Nous avons l'intention d'assister le développement d'un tel système en fournissant des formalismes et des notations pour spécifier le comportement désirable d'agents et du système multi-agent.

Ce chapitre traite la modélisation du SMA-C. Dans la première partie nous présentons une vue d'ensemble des recherches effectuées sur les SMA appliqués pour la collaboration en introduisant la notion d'agent collaboratif et quelques applications dans ce domaine. Dans la deuxième partie de ce chapitre nous proposons un formalisme pour la collaboration utile pour la spécification de la collaboration. La troisième partie est dédiée à la modélisation du SMA-C en présentant notre modèle d'agent collaborateur. Ce modèle combinant le formalisme des systèmes multi-agents avec les requis d'un collecticiel. Nous notons que le SMA-C n'est pas un collecticiel mais un système de collaboration entre agents logiciels. Après sa validation, il est utilisé comme noyau d'un collecticiel pour la téléopération.

2.2 Agent collaboratif

Plusieurs systèmes de téléopération collaborative ont été développés mais ils ont pour inconvénient d'offrir une architecture fermée et souvent spécifique à chaque type d'application. La conjugaison de l'état de l'art dans les domaines des SMA et des environnements collaboratifs (collecticiels) permet de résoudre ce problème. Sachant que la problématique de l'interaction est largement traitée dans le domaine des SMA [FER 95].
L'objectif de ce chapitre est de donner une vue d'ensemble sur les recherches sur les SMA appliqués au TCAO. Dans la première partie de ce chapitre, nous commencerons par introduire la notion des SMA dans la collaboration et présenter quelques modèles de base de ces systèmes. Ensuite, nous exposerons quelques applications dans ce domaine.

2.2.1 Collaboration versus SMA

Les dernières tendances dans les technologies de la collaboration distribuée et mobile permettent aux gens de se déplacer à travers les limites d'organisation et de collaborer avec les autres dans une même organisation ou entre différentes organisations et communautés. La capacité d'interroger la base de connaissances distribuée et de coopérer avec les collègues est encore une exigence, mais des nouveaux paradigmes tel que l'informatique orientée services (par exemple Web Services), la diffusion augmentée, et la mobilité permettent de nouveaux scénarios et mènent à plus haute complexité des systèmes.
Les architectures des logiciels des communautés coopératives distribuées et mobiles doivent supporter les exigences fondamentales pour la coopération distribuée: partage d'information effectif à travers un environnement largement distribué; mise à jour constante et opportune et placement de la base distribuée de connaissances avec différents sites qui fonctionnent, à la fois, comme des utilisateurs potentiels et comme des fournisseurs potentiels d'informations; accès partagé à un ensemble de services. Les approches et technologies pour supporter cette nouvelle façon de travailler sont encore des sujets de recherche. Néanmoins, ils empruntent des concepts et des technologies d'une variété de domaines, tel que les systèmes du workflow, les CSCW, les systèmes basés sur les événements, l'architecture des logiciels, les systèmes de base de données distribuées, l'informatique mobile, et ainsi de suite. Une ligne particulièrement intéressante de recherche explore un paradigme peer-to-peer enrichi avec le partage des abstractions

dans chaque nœud du réseau qui est, à la fois, un utilisateur potentiel et un fournisseur d'information pour le reste de la communauté collaborative. Dans le passé, les gens ont collaboré entre eux principalement à travers l'échange des informations. Avec le début de telles technologies comme les services du web sémantique, les architectures de services (SOA), et les améliorations de la bande passante, les collaborateurs se sont rendus compte de la flexibilité de collaborer par l'échange de services universellement accessibles. Cependant, avec cette flexibilité vient l'augmentation de complexité. L'autonomie et l'intelligence des agents logiciels peuvent être appliquées dans ce domaine et la complexité des collaborations basée sur les services peut être amoindrie.
L'autonomie et l'intelligence des agents logiciels ont fortement augmenté l'automatisation dans beaucoup de domaines opérationnels. Un avantage majeur de l'utilisation des agents est leur capacité d'aider, dans la collaboration, des êtres humains et des mécanismes logiciels, et ce de la même façon.

2.2.1.1 Les modèles d'agent collaboratif

Dans le comportement coopératif normatif, les normes peuvent être implicitement définies à travers le comportement des agents et dépendent de la façon selon laquelle les agents fonctionnent. Dans la stratégie de coopération, les normes sont définies explicitement et les agents doivent se conformer à ses normes.
Dans [KHA 04], Khalil a étudié le problème d'application des normes dans les institutions électroniques. Les institutions électroniques peuvent être considérées comme l'équivalent des agents dans les organisations humaines, en particulier avec la considération du support, de la confiance, et de la légitimité dans les applications du commerce électronique. Les auteurs proposent un algorithme, pour contrôler l'approbation des normes, basé sur des règles de substitution. De cette manière, les interactions des agents peuvent être contrôlées et les violations possibles des normes peuvent être sanctionnées. Cela peut causer des conflits entre les buts des agents et les buts des institutions électroniques. D'une part, si les agents doivent se conformer aux normes, les opportunités, que ces derniers peuvent avoir pour réaliser leurs buts, diminuent. D'autre part, si les agents choisissent de violer les normes pour atteindre leurs objectifs les institutions électroniques deviennent imprédictibles.
Dans [CAB 04], l'auteur présente une approche pour la communication

basée sur l'utilisation des agents mobiles. Cette approche a deux avantages principaux: le premier, un message devient une entité active, qui permet le filtrage du contenu et qui laisse le message lui-même décider proactivement de ce qu'il doit faire avec son contenu et comment le montrer à l'utilisateur final. Le second avantage est le fait que cette approche peut être intégrée dans des applications de communication existantes. Par conséquent cela facilite son exploitation par les différents utilisateurs, et permet aussi d'unifier plusieurs systèmes de message en un seul.

[STE 04] propose une plateforme pour le développement des applications d'agents. La plateforme, appelé eXAT (erlang eXperimental Agent Tool), permet d'intégrer la conception de l'intelligence d'un agent, de son comportement et de sa communication. eXAT exploite Erlang, un langage similaire au Prolog, pour définir les caractéristiques d'un agent. Etant donné ces caractéristiques, la plateforme eXAT convient à l'implémentation des applications basées sur des agents collaboratifs, en prenant en compte les différents aspects du développement d'un agent.

Les auteurs de l'article [SEL 04] présentent leurs travaux sur la collaboration des groupes assistée par des agents. Les auteurs introduisent une plateforme appelée Eksarva qui supporte la modélisation formelle de la collaboration comme une fonction de comportement et des règles du workflow. Cette approche de modélisation est utilisée pour programmer des agents qui exécutent des activités intégrées, de gestion de la connaissance, nécessaires pour une collaboration efficace du groupe.

[FRE 03] propose l'utilisation d'un système multi-agent pour la gestion de la chaîne de provision en respectant quelques perspectives comme l'organisation, le contrôle, et l'exécution. En plus, cette approche considère des rapports non fonctionnels tels que la flexibilité et la précision. Elle incorpore l'utilisation des techniques de modélisation de l'industrie standard qui utilisent l'UML. Il y a aussi une évaluation positive de l'approche basée sur l'impact sur le temps du débit.

[KIR 03] présente une architecture est une plateforme basée agents. Elle est dédiée pour supporter la collaboration dans les applications distribuées dans le domaine des soins médicaux. Les auteurs ont décrit chacun des composants de cette plateforme. Ils ont par la suite démontré la pertinence de certaines caractéristiques des agents au contexte spécifique au domaine de la santé.

Dans [CAB 03] une application basée sur un agent mobile a été proposée. Elle est exploitée pour étudier les différents types d'interactions d'agent. De telles interactions sont gérées, séparément de la logique de l'agent, au

moyen de rôles. Dans l'article, il présente aussi les avantages d'une telle approche. Celle-ci permet le développement des applications basées sur des agents flexibles (et réutilisables au niveau de l'entreprise) pour exécuter des tâches automatiques ou administratives.
[MAA 03], dans leur article, présentent l'utilisation d'agents logiciels pour la spécification d'un environnement Web de services. Leur idée est que les utilisateurs peuvent exploiter des agents pour composer des services Web dans des processus d'affaire de haut niveau. Ils proposent trois niveaux de spécification: intrinsèque, organisationnel/fonctionnel, et comportement. Les trois niveaux sont appliqués à des agents et à des services de façons différentes. Durant l'*agentification* des services, deux types d'agents logiciels sont proposés : agent-utilisateur et agent-fournisseur, chacun des deux types est associé à un domaine dédié.

2.2.2 Quelques applications d'agents collaboratifs

2.2.2.1 Le projet Collaborator

Le projet Collaborator [1] a commencé au début Novembre 2001. Le but majeur du projet est la réalisation d'un système "Collaborator". Ce système devra être capable de fournir un espace de travail partagé supportant des activités d'équipes virtuelles (figure 2.1). Un autre but du projet est d'aménager un environnement d'essai pour explorer les avantages de l'intégration de ce logiciel avec les technologies émergeantes.
En résumé, le Collaborator est prévu pour supporter le travail collaboratif à distance des équipes virtuelles rencontrant les contraintes suivantes:

- Indépendance de la plateforme et intégration avec Web: Le Collaborator est basé sur des technologies Web standards (Java, HTML, TCP/IP, etc.) et son système d'exploitation est indépendant du réseau;

- Accessibilité continuelle : Le Collaborator peut être accédé à partir d'ordinateurs de bureau et de portable et ce de façon homogène.

- Faculté d'adaptation aux bandes passantes réseaux et des capacités des terminaux: l'accessibilité permanente exige que le Collaborateur s'adapte aux capacités des terminaux et des connections que les utilisateurs peuvent utiliser pendant une réunion virtuelle;

[1] http://www.ist-collaborator.net

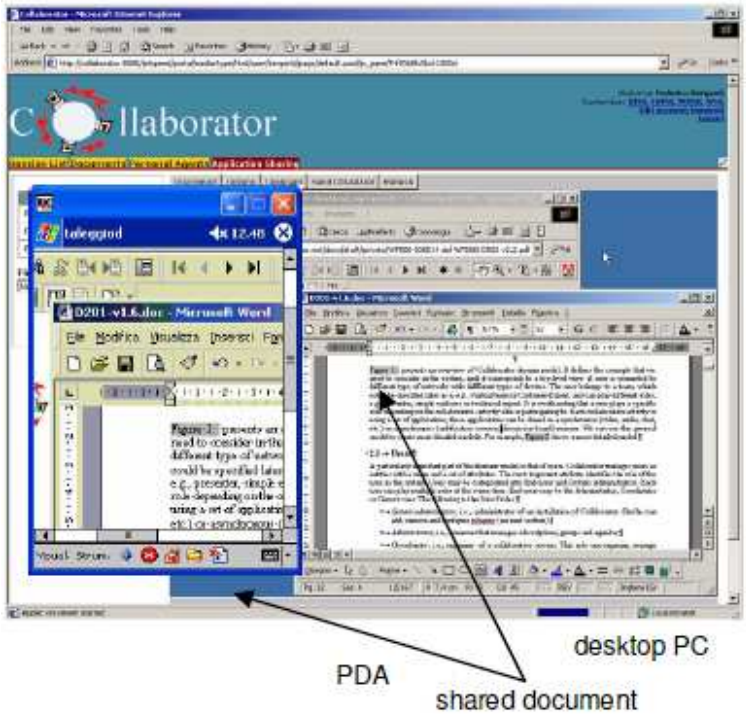

Figure 2.1: Les différentes vues du même outil partagé dans Collaborator

- Interaction flexible avec l'utilisateur à travers les agents: les personnes impliquées dans une réunion virtuelle sont associés avec des Agents Personnels qui servent de médiateurs entre leurs interactions et fournissent la personnalisation basée sur les profils.

Le résultat du projet Collaborator est un prototype d'un système qui respecte toutes ces contraintes. Ce système est conçu autour de l'idée de session. Une session est la vue dont on dispose lors d'une réunion virtuelle. Les participants utilisent leurs appareils pour participer à la réunion, et par conséquent une session est un ensemble d'activités (collaboratives) à travers un ensemble dynamique d'appareils qui utilisent plusieurs outils (par exemple, applications et éditeur du document) durant une réunion virtuelle.
Ce système respecte toutes les contraintes requises pour le travail collaboratif à distance des équipes virtuelles. Par contre il ne permet pas une interaction directe entre les différents utilisateurs. Il n'est pas non plus adapté pour la téléopération collaborative.

2.2.2.2 Le modèle de négociation collaborative

[ABD 04] propose l'utilisation des agents dans les chaînes logistiques dynamiques des entreprises virtuelles. En particulier, ils exploitent les agents coopératifs demandeurs de services dans le protocole utilisé afin de rechercher et de négocier des offres alternatives. De cette façon, la gestion flexible des offres est accomplie et les trafics florissants des services du Web sont atteints.
Les agents logiciels intelligents sont utilisés pour conduire des négociations automatiques [FAR 98], [WEI 99]. Pendant la négociation les agents prennent en considération un grand nombre de facteurs. Par exemple, le temps joue un rôle important dans la stratégie de la négociation de l'agent (la façon avec laquelle l'agent évalue et propose les offres à ses semblables). Les agents adaptent leurs stratégies pour obtenir de meilleures offres dans une date limite.
Les agents demandeurs de services coopèrent en demandant l'aide et, en retour, partagent les informations relatives aux meilleures offres du moment sous certaines conditions. Les auteurs démontrent comment l'utilisation de ce protocole peut augmenter le nombre des activités de négociation réussies dans un contexte d'acquisition de service à délais limités.
Ce système ne traite pas la dimension collaboration entre les utilisateurs.

2.2.2.3 GAP : la Plateforme d'Automatisation Globale

[GUI 04] introduit une Plateforme d'Automatisation Globale (GAP). C'est un environnement logiciel basé sur les agents, pour la mise en œuvre de systèmes d'automatisation globale. Il offre, des blocs de construction de base pour le développement des modules du contrôle (les classes d'agents de base) et les services communs exigés pour supporter leur activité. GAP est construit sur une infrastructure multi-agent, appelé G++ Agent Platform. Cette plateforme agent supporte le développement, l'exécution, et l'intégration des agents comme le font les autres plateformes commerciales ou de recherche. Elle a l'avantage d'offrir une architecture extensible pour les systèmes à grande échelle avec support spécial des aspects liés à la communication. Les auteurs introduisent un système composé d'un modèle innovateur disposé en couches de plusieurs niveaux d'abstraction. Ces niveaux d'abstraction varient des capacités des agents d'un domaine spécifique à un autre. La figure 2.2 montre les différentes couches du système d'automatisation globale de GAP. Ce système n'intègre pas les fonctionnalités de la collaboration.

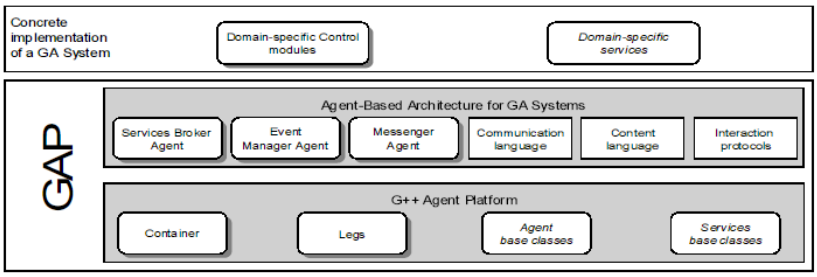

Figure 2.2: Les différentes couches du système d'automatisation globale de GAP

2.2.2.4 Une plateforme pour la collaboration

Dans son travail, Heroux [HER 04] envisage l'utilisation de sources de données pour commencer de nouvelles sessions de collaboration parmi les collaborateurs distribués. Les agents logiciels sont solidement intégrés avec des portions de données et ils gèrent la création de nouveaux réseaux collaboratifs.
Il définit une "balise de collaboration" en la décrivant ; comme étant légère, indépendante de la plateforme et comme une application agent de but spécial. Cette "balise de collaboration" contient les éléments informationnels (l'objet) sur lesquels les services des agents sont appliqués (les méthodes). Contrairement aux services du Web, qui sollicitent un faible groupement entre les informations et les services, les balises, quant à elles, requièrent une forme plus serrée d'association pour accomplir leur rôle d'agents permanents pour des instances particulières de données.

2.2.3 Bilan de l'agent collaboratif

Les modèles d'agent collaboratif, présentés dans cette partie, ne présentent que des normes et des modèles spécifiques à leurs problématiques. Ces modèles ne caractérisent, en aucun cas, la notion de collaboration, comme nous l'avons défini suite à l'étude de l'état de l'art des TCAO. En revanche, la collaboration dans ces modèles ce n'est que l'aide du système fournie pour l'utilisateur ou bien la collaboration entre les différentes entités du système lui même.
Les applications d'agent collaboratif, quant à elles, ne supportent que le travail des équipes virtuelles et donc elles ne tiennent pas compte des contraintes du monde réel, ou, dans d'autres cas, ces applications offrent des systèmes d'automatisation globale qui ne considèrent que la coordination

entre les agents au sein d'un SMA.

Dans ce qui suit, nous allons présenter une modélisation de la collaboration, comme nous l'avons défini, et qui se base sur des agents dédiés à la collaboration. En effet, pour nous, une collaboration est, d'une part, une coopération entre les membres d'une équipe et, d'autre part, une réalisation d'un produit fini. La coopération, entre les membres d'une équipe, est déterminée par une communication et une coordination entre tous les membres de l'équipe. La collaboration, est donc, définie par la communication, la coordination et la production.

2.3 Modélisation de la collaboration

Comme nous l'avons déjà vu au paragraphe 1.3, la collaboration est caractérisée par trois espaces tels que l'espace de communication, l'espace de coordination et l'espace de production. Une collaboration est un travail en commun; un travail entre plusieurs personnes qui produisent un résultat commun (produit final). Pour mener ce travail convenablement, les participants doivent se coordonner et communiquer ensemble. Pour se coordonner, les participants doivent suivre l'activité des autres participants pour l'utilisation et le partage de la ressource commune.
Généralement, la communication est considérée au sens de la conversation libre entre participants, non obligatoire et n'apportant aucun élément indispensable pour la collaboration [DAV 01]. Cependant, la coordination peut aussi avoir lieu à travers la communication entre participants et par conséquent la communication entre les différents membres de l'équipe est primordiale pour le succès du travail collaboratif. Dans notre système, nous supposons donc que la collaboration est basée sur la communication, en d'autres termes nous ne pouvons pas avoir une coordination sans communication, ni production sans coordination. Cependant, nous pouvons communiquer sans coordonner et coordonner sans produire.
Nous rappelons que notre problématique principale est de mettre en place un collecticiel pour la téléopération. Cela suppose une manipulation directe sur un robot réel et donc une tâche délicate à effectuer qui nécessite beaucoup de rigueur de la part des participants. Pour cette raison un participant doit impérativement communiquer avec les autres pour mettre en place un plan de coordination. Par conséquent la communication doit essentiellement permettre aux membres du groupe qui doivent collaborer ensemble

de se mettre d'accord sur les règles de collaboration. Ils doivent décider sur le but de la collaboration, et de la façon de l'effectuer et également sur les résultats attendus de chacun des participants. En effet, la communication dans notre système est considérée outre, au sens de la conversation libre, comme un élément obligatoire et indispensable pour l'aboutissement de la collaboration. Cela n'élimine pas la possibilité de discuter librement une fois que les participants se sont mis d'accord pour la collaboration.

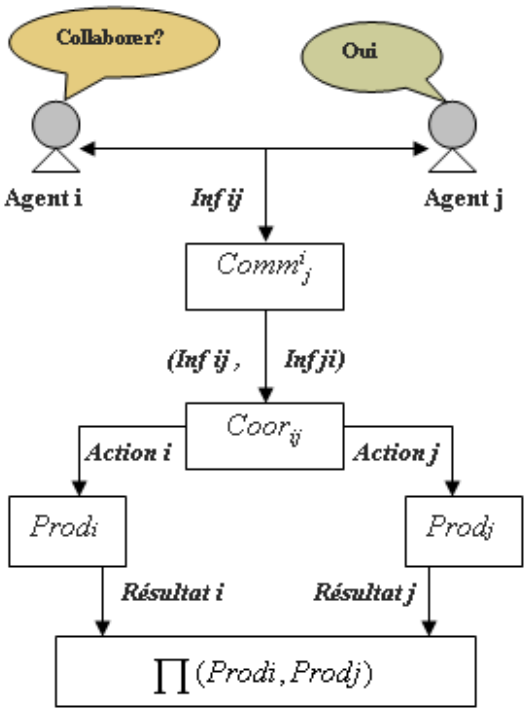

Figure 2.3: Le protocole de collaboration entre deux agents

Le formalisme de collaboration que nous proposons [KHE 04] nécessite au moins deux agents interagissant ensemble afin d'exécuter une tâche commune ou d'atteindre un but commun. Supposons qu'un agent i demande à un agent j de collaborer avec lui comme illustré dans la figure 2.3.
Dans l'équation (2.1), nous avons considéré la collaboration $Coll^i_j$ d'un agent i avec un autre agent j. Une collaboration accomplie entre un agent i et un agent j est modélisée par une communication, une coordination et les productions des deux agents.

$$Coll^i_j =< Comm^i_j, Coor_{ij}, Prod_i, Prod_j > \tag{2.1}$$

La communication est entamée par l'agent i, d'où la notation $Comm^i_j$;

l'agent i commence la communication en envoyant une demande de collaboration Inf_{ij} à l'agent j (équation (2.2)). L'agent j, dans tous les cas, envoie une réponse Inf_{ji} à l'agent i . La communication se repose sur un échange de couples d'information (Inf_{ij}, Inf_{ji}), chaque fois qu'un agent entame une discussion, il reçoit une réponse.

$$Comm^{i}_{j}(Inf_{ij}) = \{(Inf_{ij}, Inf_{ji})\} \tag{2.2}$$

Selon les accords des deux agents i et j, la coordination peut commencer (équation (2.3)), toujours, en discutant et en échangeant les couples d'information (Inf_{ij}, Inf_{ji}), les deux agents vont mettre en place des plans d'actions $(Actions_i, Actions_j)$ que chacun d'entre eux doit exécuter.

$$Coor_{ij}\left(\{Inf_{ij}, Inf_{ji}\}\right) = \{Actions_i, Actions_j\} \tag{2.3}$$

Après la phase de coordination commence la phase de production(équation (2.4)). Chacun des deux agents, par exemple l'agent i (respectivement l'agent j) exécute ses propres actions $Actions_i$ (respectivement $Actions_j$) et produit des résultats partiels $Resultats_i$ (respectivement $Resultats_j$).

$$Prod_i\left(\{Actions_i\}\right) = \{Resultats_i\} \tag{2.4}$$

Une fois les résultats partiels obtenus, ils sont combinés (à l'aide de l'opérateur de combinaison) dans le but d'avoir un résultat global de collaboration (un produit final).
Nous considérons la collaboration globale (2.5) entre N agents. La collaboration $COLL$ d'un ensemble d'agents est un triplet $COMM$, $COOR$ et $PROD$, tel que $COMM$ est la communication globale, $COOR$ est la coordination globale et $PROD$ est la production globale de tous les agents qui collaborent ensemble.

$$COLL = \langle COMM, COOR, PROD \rangle \tag{2.5}$$

La communication globale $COMM$ (équation 2.6) est représentée par tous les couples d'informations échangés entre un agent i et un autre j, pour chaque agent i qui communique avec un autre agent j.

$$COMM = \{(Inf_{ij}, Inf_{ji}) \,/ i = 1 \cdots N-1, j = 1 \cdots N \,, i \neq j\} \tag{2.6}$$

La coordination globale $COOR$ du système (équation 2.7) est représentée par l'ensemble des actions $Actions_i$ de tous les agents i collaborant.

$$COOR = \{Actions_i \, / i = 1 \cdots N\} \tag{2.7}$$

La production globale $PROD$ (équation 2.8) est la combinaison de tous les résultats partiels $Resultats_i$ de tous les agents i collaborant. Nous rappelons que l'opérateur de combinaison dépend du type de l'application.

$$PROD = \prod_{i=1 \cdots N} (Resultats_i) \tag{2.8}$$

De cette modélisation formelle de collaboration, nous pouvons retenir quelques points clés pour le reste de ce manuscrit :

- La collaboration est tout d'abord une communication, ensuite une coordination et finalement une production.
- La communication est basée sur des couples d'information; chaque demande envoyée par un agent i à un agent j doit avoir une réponse.
- Suite à la coordination, chacun des agents participants à la collaboration, définit avec les autres agents du même espace son plan d'actions, qu'il doit respecter pour le bon fonctionnement de la collaboration.
- Et puisque tous les résultats de toutes les productions seront combinés, alors il est nécessaire de respecter les contraintes, dues à cette combinaison, durant l'exécution des actions.

Dans la section suivante, nous présenterons un modèle formel d'agent collaborateur; un modèle formel d'agents qui participent à une collaboration.

2.4 Modélisation de l'Agent Collaborateur : AC

2.4.1 Formalisme de l'AC

Dans cette partie, nous présentons notre formalisme d'agent collaborateur [KHE 05c] en se basant, d'une part sur le modèle générique d'un agent présenté dans la section 1.2.4.2 et d'autre part sur le formalisme de la collaboration que nous avons proposé et présenté dans la section 2.3.

Remarque 2.1 *Il faut noter que dans ce paragraphe et éventuellement dans le reste du manuscrit, lorsque nous parlerons d'agent, nous voudrons*

dire un agent collaborateur; et lorsque nous parlerons d'un système pour les agents, nous ferons allusion à un système de collaboration et que tous les agents présents dans ce système doivent collaborer ensemble.

Nous avons repris la définition d'un agent, comme telle que présentée par Ferber, un agent i qui interagit avec l'environnement peut être considéré comme un couple de systèmes dynamiques $< i, w >$. Ferber dans son système, modélise deux états de l'agent un état interne et un état global. Dans notre modélisation, nous ne considérons qu'un seul état, qui est son état global vis à vis à la tâche de collaboration.

$$i =< P_i, Percept_i, F_i, Infl_i >$$
$$w =< E, \Gamma, \Sigma, R >$$

Un monde w d'un agent est bien évidemment, dynamique et changeant à chaque action ou réaction de l'agent collaborateur. Ce monde est modélisé par un environnement E, un ensemble de répercussions Γ causées par l'agent en question suite à la transformation de son environnement, un ensemble d'états Σ par lesquels l'agent collaborateur passe et d'une loi R d'évolution du monde qu'il doit respecter pour pouvoir interagir et coopérer avec ses équivalents.
Notre agent collaborateur évolue dans un monde conçu pour la collaboration, donc, les définitions des différents éléments doivent intégrer les fonctionnalités de la collaboration, à savoir, la communication, la coordination et la production.

L'environnement E de l'agent collaborateur, l'espace dans lequel il évolue et qui est représenté par l'ensemble des agents collaborateurs de l'environnement collaborant avec cet agent.

L'ensemble Γ des actions produites par l'agent collaborateur et ayant comme conséquences de modifier l'évolution du monde, dans notre cas, Γ sera l'ensemble de trois sous ensembles :

- $\{Inf_{ij}\}$: l'ensemble des informations envoyées par l'agent i vers les autres agents j collaborants de l'environnement.
- $\{Actions_i\}$: l'ensemble des actions exécutées par l'agent i.
- $\{Resultats_i\}$: l'ensemble des résultats issus de l'exécution des actions de l'agents i.

L'ensemble Γ est alors représentée par la fonction suivante :

$$\Gamma = \{\{Inf_{ij}\}, \{Actions_i\}, \{Resultats_i\}\}$$

L'ensemble Σ est l'ensemble des états de l'agent collaborateur, en ce qui concerne notre modélisation, un agent collaborateur i peut avoir trois états : communication ($comm$), coordination ($coor$) ou production ($prod$), et ce comme l'indique l'équation suivante :

$$\Sigma = \{comm, coor, prod\}$$

Nous pouvons tracer un graphe d'états pour chaque agent collaborateur en interaction avec son environnement. Soit un agent collaborateur i, son graphe d'états est présenté dans la figure 2.4. Dans cette figure, l'état de départ est lorsque l'agent collaborateur i est crée (le disque noir), il sera mis tout de suite en état d'attente d'action. Cet état ne sera pas pris en compte dans notre modèle formel car c'est un état transitoire et temporaire durant lequel l'agent collaborateur i ne fait aucune action.

Par la suite, cet agent (collaborateur i) va envoyer ou recevoir une demande de collaboration (nous détaillerons cet étape dans le chapitre suivant) et dans ce cas son état va transiter vers l'état de communication. Si la communication est réussie alors l'agent passe à l'état de coordination, signalant que les conditions de réussite dépendent de l'application envisagée (un exemple est détaillé dans le chapitre suivant). Par contre si la communication échoue, nous retrouvons deux cas possibles. Dans le premier cas, la communication échoue et il n'y a plus d'autre collaborateurs à solliciter, alors l'agent passe à l'état fin (il se tue, par exemple, cet état est représenté sur le schéma par le disque noir encerclé). Dans le deuxième cas, si la communication échoue et il y a d'autres collaborateurs susceptibles de collaborer avec lui, alors l'agent i transite vers l'état d'attente et recommence la procédure dès le début.
Si la communication est réussie (c'est-à-dire l'agent i a réussi à se mettre d'accord avec un ou plusieurs autres agents pour collaborer), l'agent i passe à l'état coordination, dans lequel il va définir, avec les autres participants de la collaboration, ses plans d'actions. Comme pour la communication, quand la coordination échoue, deux cas de figures peuvent se présenter. Le premier, si la coordination échoue

et il n'y a plus d'autres agents collaborateurs avec lesquels l'agent i pourra collaborer, alors il passe à l'état fin. Dans le deuxième cas, si la coordination échoue et il y a d'autres collaborateurs susceptibles de collaborer avec lui, alors l'agent i transite vers l'état d'attente et recommence la procédure dès le début.

Si la coordination est réussie, chacun des agents participants à la collaboration a pu définir, en se coordonnant avec ses homologues, ses actions qu'il doit exécuter selon les conditions prédéfinies par l'application. Dans ce cas, chacun des agents participants à la collaboration est engagé à exécuter ses actions individuellement.

Dans l'application de ce modèle sur un exemple concret présenté dans le chapitre suivant, nous pourrons constater que la non communication des différents agents collaborateurs pendant la production, n'exclut pas un échange d'information nécessaire à la production (s'il existe des contraintes de productions).

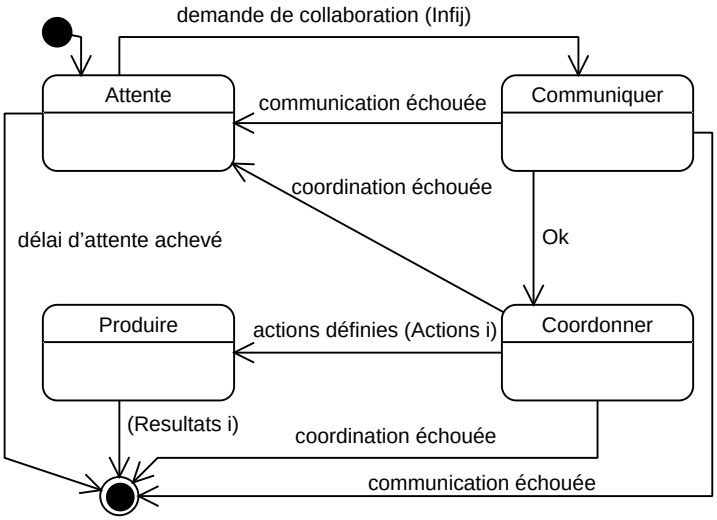

Figure 2.4: Les différents états d'un agent collaborateur i

Donc un agent i ne peut pas :

coordonner sans communiquer, et

produire sans coordonner.

Mais il peut tout de même

coordonner sans produire, ou

communiquer sans coordonner.

La loi R devient la loi d'évolution de la collaboration. Ainsi, pour qu'une collaboration soit réussie, il faut que chaque agent collaborateur i respecte cette loi. Nous rappelons qu'un agent change d'état suite aux influences qu'il produit, en d'autres termes suite aux informations qu'il envoie aux autres agents, soit pour répondre à une requête d'un autre agent, soit en prenant des décisions et en faisant des choix qui influencent les perceptions des autres agents. Par conséquent, cette loi définie par l'équation suivante, ne change pas par rapport à la définition donnée par Ferber :

$$R : \Sigma \times \Gamma \rightarrow \Sigma$$

Si nous remplaçons chacun des éléments de l'équation par sa nouvelle définition en respectant les caractéristiques de la collaboration, la loi R aura l'aspect de la fonction ci-dessous.

$$R : \{comm_i, coor_i, prod_i\} \times \{\{Inf_{ij}\}, \{Actions_i\}, \{Resultats_i\}\} \rightarrow \{comm_i, coor_i, prod_i\}$$

Cette équation exprime qu'un agent collaborateur i étant dans un état donné, suite aux informations qu'il apporte, soit il change d'état, soit il reste dans le même état. Par exemple, si un agent collaborateur i est dans un état de communication, il envoie des messages à un ou plusieurs autre agents collaborateurs du système. Il reçoit ensuite des réponses si ces réponses le satisfont, alors il passe à l'état de coordination. Dans le cas où les réponses ne répondent pas à ses attentes il demeure dans un état de communication.

Un agent collaborateur i est modélisé par quatre paramètres, sa fonction de perception P_i, son ensemble de perceptions reçues $Percept_i$, sa fonction de comportement suite à ces données F_i et enfin sa fonction de production des influences suite à son comportement $Infl_i$.
Notre agent est destiné à la collaboration, en plus, des propriétés d'agent qu'il doit avoir, nous devons tenir compte, lors de sa définition, des exigences de la collaboration. Par conséquent, les définitions des quatre paramètres

de l'agent intègrent les fonctionnalités de la collaboration, à savoir, la communication, la coordination et la production.

L'ensemble $Percept_i$ est l'ensemble des stimuli et des sensations qu'un agent collaborateur peut recevoir. En d'autres termes, c'est l'ensemble des informations apportées par les autres agents collaborateurs du système. Cet ensemble est composé des trois sous ensembles :

- $\{Inf_{ji}\}$: l'ensemble des informations reçues par l'agent collaborateur i.
- $\{Actions_j\}$: l'ensemble des actions exécutées par les agents collaborateurs j de l'environnement, autre que l'agent collaborateur i.
- $\{Resultats_j\}$: l'ensemble des résultats produits par les agents collaborateur j de l'environnement, autre que l'agent collaborateur i.

Par conséquent, l'ensemble $Percept_i$ adapté à la collaboration, se définit par l'équation ci-dessous.

$$\forall j,\ Percept_i = \{\{Inf_{ji}\}, \{Actions_j\}, \{Resultats_j\}\}$$

En effet, chaque activité (informations envoyées, actions définies ou résultats produits) d'un autre agent collaborateur j du système influence directement cet agent.

La fonction P_i représente la fonction de perception de l'agent. C'est la fonction qui permet à un agent de cerner l'ensemble des informations qui conditionnent son comportement et qui déterminent l'ensemble $Percept_i$ mentionné ci-dessus.

$$P_i : \Sigma \rightarrow Percept_i$$

Nous intégrons les caractéristiques de la collaboration dans la fonction ci-dessus, nous obtenons :

$$\forall j,\ P_i : \{comm_i, coor_i, prod_i\} \rightarrow \{\{Inf_{ji}\}, \{Actions_j\}, \{Resultats_j\}\}$$

Cette équation montre qu'un agent collaborateur i, étant dans un état donné, reçoit des influences produites par les autres agents collaborateurs du système.

La fonction F_a est la fonction de comportement de l'agent qui détermine l'état de l'agent à partir de ses perceptions et de son état précédent,

$$F_i : \Sigma \times Percept_i \rightarrow \Sigma$$

En intégrant les fonctionnalités de collaboration à cette équation, nous obtenons :

$$\forall j, \quad F_i : \{comm_i, coor_i, prod_i\} \times \{\{Inf_{ji}\}, \{Actions_j\}, \{Resultats_j\}\} \rightarrow \{comm_i, coor_i, prod_i\}$$

Un agent collaborateur i, étant dans un état donné, reçoit des influences produites par les autres agents collaborateurs du système, peut changer d'état. Comme nous l'avons expliqué un peu plus haut, il peut garder le même état.

La fonction $Infl_i$ est la fonction d'action de l'agent collaborateur qui tend à modifier l'évolution du monde en produisant des influences. Un agent collaborateur, étant dans un état donné et suite à cet état il peut produire des informations qui changent son évolution et l'évolution des autres agents collaborateurs présents dans le système.

$$Infl_i : \Sigma \rightarrow \Gamma$$

où

$$Infl_i : \{comm_i, coor_i, prod_i\} \rightarrow \{\{Inf_{ij}\}, \{Actions_i\}, \{Resultats_i\}\}$$

Considérons un agent collaborateur i de l'environnement E et σ, un état de cet agent, un élément de Σ. L'état de l'agent collaborateur i à un instant $(t+1)$ est calculé en fonction de son état précédent, à l'instant t, et de ses perceptions :

$$\sigma(t+1) = F_i(\sigma(t), P_i(\sigma(t)))$$

L'évolution de l'état de l'agent i est donnée par l'équation suivante :

$$\sigma(t+1) = R(\sigma(t), Infl_i(\sigma(t)))$$

Un système multi-agent pour la collaboration est défini par un triplet $< A, W, \Pi >$ où A est un ensemble d'agents collaborateurs, W un monde et Π un opérateur de combinaison d'influences. La dynamique du système est alors donnée par le système des $n+1$ équations suivantes:

$$\begin{aligned}
\sigma(t+1) &= F_1\left(\sigma\left(t\right), P_1\left(\sigma\left(t\right)\right)\right) \\
&\cdots \\
\sigma(t+1) &= F_n\left(\sigma\left(t\right), P_n\left(\sigma\left(t\right)\right)\right) \\
\sigma(t+1) &= R\left(\sigma\left(t\right), \prod_{i=1\cdots n} Infl_i\left(\sigma\left(t\right)\right)\right)
\end{aligned}$$

Pour un système multi-agent composé de n agents, l'état de chaque agent σ, à un instant $t+1$, dépend de son état σ à l'instant t et de son évolution dans son environnement, en suivant la même loi que suivent les autres agents du système. Soit un système multi-agent pour la collaboration, supposons que tous les agents présents dans le système collaborent pour atteindre un but commun, alors l'état de chaque agent est le même pour tout le système. Par conséquent, tous les agents ont le même état à un instant donné. Cela découle directement de la définition même de la collaboration, à savoir la communication puis la coordination et enfin la production. Donc, tout le monde communique (le même état *comm*) et quand ils trouvent un arrangement, ils commencent la coordination (l'état *coor*) et enfin ils produisent (l'état *prod*). Par conséquent, l'état global du système à l'instant $t+1$ évolue en fonction de l'état précédent de tous les agents collaborateurs présents dans le système à l'instant t et de la combinaison des différentes influences produites par tous les agents du système.
Il est primordial dans un système multi-agent, que tous les agents suivent la même loi d'évolution, c'est-à-dire qu'ils ont les mêmes règles au sein du même groupe. De ce fait, la société d'agents collaborateurs repose sur une loi commune à tous les agents collaborateurs.

2.4.2 Modèle fonctionnel de l'AC

L'architecture proposée a pour objectif l'aide à la collaboration ainsi que la résolution des conflits qui en résultent. L'utilisation des agents permet

d'instaurer un environnement flexible (agents capables de répondre à temps, pro-actifs, communiquants) dans le but de l'exécution ordonnée et concluante des tâches jusqu'à la terminaison de la mission en des bonnes conditions. Le système multi-agent défini est composé de quatre types d'agents (figure 2.5): les agents qui permettent de choisir une mission ; appelés les **agents Communication**, les agents qui permettent de sélectionner les actions à exécuter ; appelés les **agents Coordination**, les agents qui exécutent les actions choisies, appelés les **agents Production**, et les agents qui permettent d'assurer la communication inter agents, appelés les **agents Collaboration**. Chaque instance de ces quatre agents est regroupée sous un super agent, appelé l'**agent Collaborateur**.

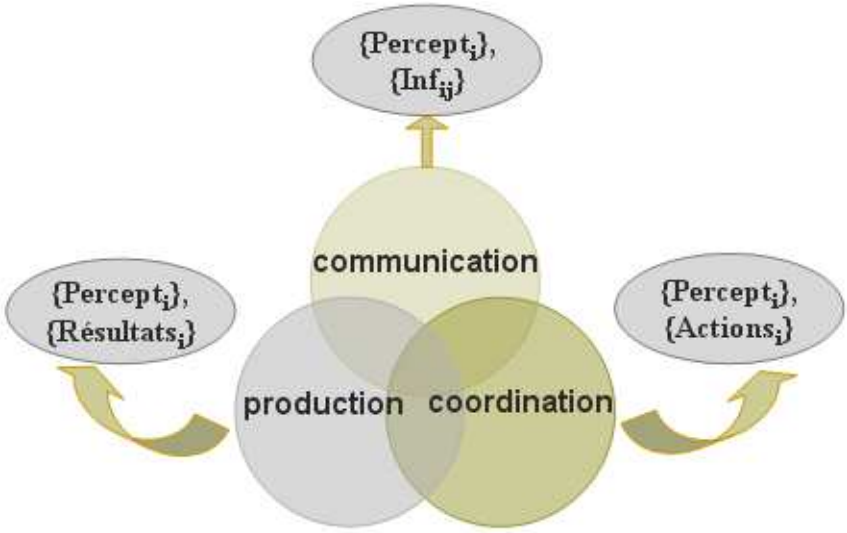

Figure 2.5: Modèle fonctionnel de l'agent collaborateur

Dans ce modèle d'architecture, un agent collaborateur interagit avec ses correspondants du système par l'intermédiaire de son agent collaboration et de ses agents dédiés (communication, coordination et production) à l'exception que chacun de ces agents communique avec ses homologues des autres agents collaborateurs. Cela veut dire, qu'un agent communication d'un agent collaborateur i, par exemple, peut communiquer avec l'agent communication de l'agent collaborateur j idem pour l'agent coordination i avec l'agent coordination j.

Dans ce contexte, nous faisons la différence entre l'agent communication, lequel son rôle est d'aider les agents collaborateurs pour se mettre d'accord sur la mission à exécuter, et la communication entre les différents agents (ou inter agents) qui est la communication de bas niveau. Cette communication

de bas niveau permet l'interaction entre les différents agents et nous pouvons citer comme exemple de ce type de communication, l'envoi d'un message ACL (Agent Communication Language) d'un agent vers un autre.

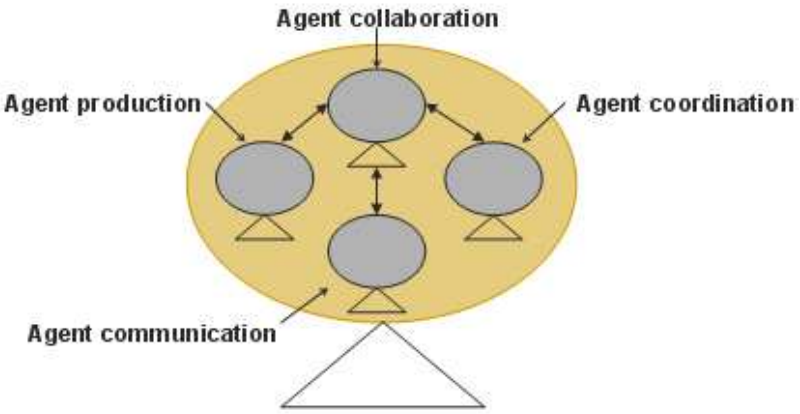

Figure 2.6: La description de l'agent collaborateur

Dans notre modèle, chaque agent collaborateur (figure 2.6) sera alors composé de trois agents dédiés respectivement à la *communication*, à la *coordination* et à la *production*, le quatrième agent collaboration va jouer le rôle d'intermédiaire entre ces trois agents dédiés. Ce choix est justifié par le fait que nous voulons limiter le nombre d'échange de messages de chaque agent dédié.
L'avantage majeur de cette architecture c'est l'utilisation de l'agent collaboration qui joue un double rôle, d'une part, il assure la liaison entre les différents agents du monde considéré et d'autre part il est vu comme une interface entre l'utilisateur et l'agent collaborateur.

2.4.2.1 Agent collaboration

L'agent collaboration assure deux fonctions principales. D'une part, il permet la communication entre les trois agents dédiés de l'agent collaborateur et d'autre part, il établit une interaction directe entre chaque agent dédié avec son correspondant d'un autre agent collaborateur, ceci leurs permet de communiquer ensemble sans l'intervention de l'agent collaboration. Cette communication directe entre les agents dédiés, d'un même agent collaborateur, et avec leurs correspondants des autres agents collaborateurs, permet d'augmenter l'efficacité du système de communication. Cela permet d'éviter l'effet du goulot d'étranglement dû à la centralisation. En fait, les messages

envoyés par les différents agents ne concernent pas seulement la communication. Le seul moyen d'interaction des agents entre eux dans notre système est l'envoi des messages. Par conséquent, les messages de communication de bas niveau (inter agents) sont échangés entre les agents durant toutes les phases de la collaboration. Par contre, l'agent collaboration assure la communication inter agents dédiés d'un même agent collaborateur, qui eux ne communiquent pas directement et ce pour limiter le nombre de messages échangés entre ces agents dédiés et aussi pour éviter la perte d'information. En effet, quand l'agent communication finit son travail, il communique ses résultats à l'agent collaboration. Ce dernier se charge d'informer l'agent coordination de commencer ou pas son travail. Idem pour l'agent coordination et pour l'agent production.

2.4.2.2 Agent communication

L'agent communication décide si une mission peut être accomplie ou pas, et c'est donc lui qui décide si la collaboration peut avoir lieu. Par conséquent, l'agent communication est la pierre angulaire de l'agent collaborateur. L'agent communication détermine l'état de l'agent en fonction des perceptions qu'il reçoit et de son état antérieur. Cet agent gère toutes les informations qui lui sont communiquées. Ces informations représentent les différentes perceptions qu'il peut recevoir de l'extérieur ($Percept_j$), comme celles émises par l'agent collaboration (Inf_i). L'agent communication définit son protocole d'interaction avec les autres agents et ce protocole est choisi selon le type de l'agent destination qui peut être extérieur ou intérieur :

- Agent extérieur : dans ce cas, l'agent communication est en interaction avec l'agent communication d'un autre agent du monde.
- Agent intérieur : dans ce deuxième cas, l'agent destination est l'agent collaboration.

L'ensemble des entrées et sorties de l'agent communication sont :

- $\{Inf_{ij}\}$: l'ensemble des informations envoyées par l'agent communication i vers les autres agents communication j avec lesquels il est en collaboration.
- $\{Percept_i\}$: l'ensemble des stimuli et des sensations qu'il peut produire.

2.4.2.3 Agent coordination

L'agent coordination définit toutes les actions que l'agent peut accomplir suite au choix de la mission de l'agent communication. L'agent coordination reçoit un ensemble d'informations qui lui sont envoyées par les autres agents coordination ou par l'agent collaboration. Cet ensemble regroupe les différentes perceptions du monde reçues par celui-ci ($Percept_j$) ainsi que les informations qui lui sont envoyées par l'agent collaboration (Inf_i). Comme pour l'agent communication, l'ensemble des entrées et sorties de l'agent coordination sont:

- $\{Actions_i\}$: l'ensemble des actions exécutées par l'agent i.
- $\{Percept_i\}$: l'ensemble des stimuli et des sensations qu'il peut produire.

2.4.2.4 Agent production

L'agent production se charge de l'exécution des actions issues de la collaboration des différents agents du système. Ces actions, sont envoyés par l'agent collaboration. L'agent production reçoit les informations qui lui sont envoyées par l'agent collaboration (Inf_i). Comme pour les deux autres agents, l'ensemble de ses entrées et sorties sont :

- $\{Percept_i\}$: l'ensemble des stimuli et des sensations qu'il peut produire.
- $\{Rsultats_i\}$: l'ensemble des résultats issus de l'exécution des procédures associées aux ses actions.

2.4.3 Les interactions entre les différents ACs

Dans notre modèle conceptuel, chaque agent peut communiquer avec un autre agent en utilisant un protocole d'interaction conforme aux spécifications de FIPA et fourni par JADE. La FIPA spécifie un ensemble de protocoles d'interaction qui peuvent être utilisés comme gabarits standards pour construire les conversations entre les agents. Pour chaque conversation entre agents, JADE distingue le rôle de l'Initiator (l'agent qui commence la conversation) et le rôle Responder (l'agent qui prend part à une conversation après qu'il soit contacté par un autre agent).
Les figures 2.7 et 2.8 montrent le processus de collaboration entre deux agents collaborateurs (l'agent collaborateur i et l'agent collaborateur j).

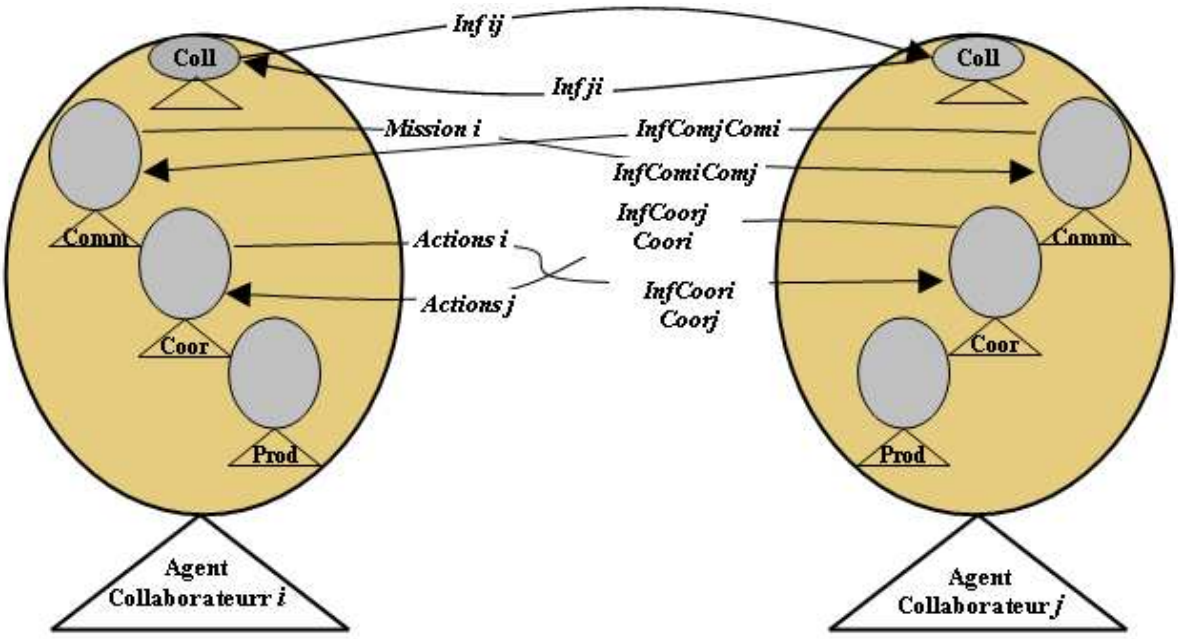

Figure 2.7: Interactions externes entre deux agents durant un processus de collaboration

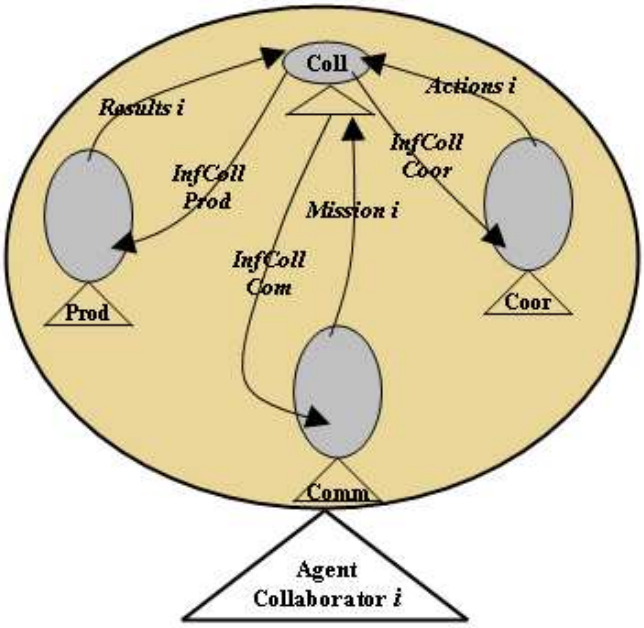

Figure 2.8: Interactions internes entre deux agents durant un processus de collaboration

L'agent collaborateur i est l'agent qui entame le processus de la collaboration. Donc, il choisit, selon les critères de choix de l'application, un autre collaborateur. Ensuite il commence la collaboration (Algo 2.9), le processus de choix d'un collaborateur n'est pas représenté dans cet algorithme. Une fois qu'il a choisi un collaborateur j, il lui envoie la demande de collaboration (par le biais de l'agent collaboration). Si l'agent collaboration j accepte de collaborer alors il envoie un message de confirmation avec lequel il envoie l'adresse de ses deux agents communication j et coordination j. L'agent communication i communique alors avec l'agent communication j pour choisir une mission (Algo 2.10). Après cela, l'agent coordination i communique avec l'agent coordination j pour assigner les actions de chaque agent (Algo 2.11). Finalement, les agents production i et production j exécutent les actions définies précédemment (Algo 2.12).
Dans les figures 2.7 et 2.8, toutes les informations entre agents commencent par Inf (par exemple, les informations échangées entre deux agents communication sont représentées par Inf_{ij} si l'agent communication i est l'émetteur et l'agent communication j est le receveur). $Mission_i$ est la mission choisie par l'agent communication i et l'agent communication j (ligne 6 d'Algo 2.10). $Actions_i$ et $Actions_j$ sont les actions définies par l'agent coordination i et l'agent coordination j (ligne 7 d'Algo 2.11). $Resultats_i$ sont les confirmations d'exécution de l'agent production i (ligne 6 d'Algo 2.12).

2.5 Bilan

Dans ce chapitre, nous avons présenté un modèle d'un système multi-agent basé sur le formalisme d'agent collaborateur proposé. Ce modèle est inspiré du modèle PAC*, et en tenant compte des propriétés présentées ci-dessous. Chaque agent de notre SMA-C est appelé un agent collaboratif. Ce dernier est composé de quatre sous-agent : l'agent collaboration (équivalent de l'agent ciment de PAC*), un agent communication, un agent coordination et un agent production. Nous avons mis en place un modèle d'interaction spécifique pour ces agents dans lequel nous distinguons les interactions internes (les interactions des quatre sous-agents de l'agent collaborateur) et les interactions externes (les interactions inter agents collaborateurs). Ce modèle tient compte des propriétés propres aux systèmes multi-agents et des caractéristiques de la collaboration.

```
1.  Collaboration ( Coll_i) :
2.      tant que (MessageQueue NON vide) faire
3.          message ⟵ receiveMessage();
4.          si (message.sender = Coll_j) alors
5.              AdressComm_j ⟵ message.getContent(1);
6.              AdressCoor_j ⟵ message.getContent(2);
7.              send (AdressComm_j, Comm_i);
8.          si (message.sender = Comm_i) alors
9.              Mission_i ⟵ message.getContent();
10.             send (AdressCoor_j, Coor_i);
11.             send (Mission_i, Coor_i);
12.         si (message.sender = Coor_i) alors
13.             Actions_i ⟵ message.getContent();
14.             send (Actions_i, Prod_i);
15.         si (message.sender = Prod_i) alors
16.             Results_i ⟵ message.getContent();
17.             send (Results_i, Coll_j);
```

Figure 2.9: L'algorithme de collaboration d'un agent $Coll_i$

```
1. Communication ( Comm_i) :
2.     tant que (MessageQueue NON vide) faire
3.         message ⟵ receiveMessage();
4.         si (message.sender = Coll_i) alors
5.             Comm_j ⟵ message.getContent();
6.             Mission_i ⟵ ChoiceMissionWith(Comm_j);
7.             send (Mission_i, Coll_i);
```

Figure 2.10: L'algorithme de collaboration d'un agent $Comm_i$

```
1. Coordination ( Coor_i) :
2.    tant que (MessageQueue NON vide) faire
3.       message ⟵ receiveMessage();
4.       si (message.sender = Coll_i) alors
5.          Coor_j ⟵ message.getContent(1);
6.          Mission_i ⟵ message.getContent(2);
7.          Actions_i ⟵ ChoiceActionsWith(Coor_j, Mission_i);
8.          send (Actions_i, Coll_i);
```

Figure 2.11: L'algorithme de collaboration d'un agent $Coor_i$

```
1. Production ( Prod_i) :
2.    tant que (MessageQueue NON vide) faire
3.       message ⟵ receiveMessage();
4.       si (message.sender = Coll_i) alors
5.          Actions_i ⟵ message.getContent();
6.          Results_i ⟵ ExecuteActions(Actions_i);
7.          send (Results_i, Coll_i);
```

Figure 2.12: L'algorithme de collaboration d'un agent $Prod_i$

(a) Les propriétés SMA :

- *La communication*: les agents communiquent ensemble pour trouver un arrangement. Cette propriété est commune au SMA et au systèmes de collaboration.
- *L'interaction*: les agents ont un protocole d'interaction bien défini. Si un agent reçoit une information d'un autre agent, il est dans l'obligation de répondre.
- *La coordination*: les agents se coordonnent pour collaborer. Cette propriété, aussi, est commune.
- *L'intelligence sociale*: les agents sont autonomes, chacun d'entre eux prend une décision (de coordonner ou pas) selon les informations dont il dispose.

(b) Les caractéristiques Collaboration :

- *La communication*: les agents communiquent pour choisir une mission. c'est une propriété commune avec les SMA.
- *La coordination*: c'est une propriété commune avec les SMA.
- *La production*: les agents quand ils s'engagent dans un processus de collaboration et qu'ils définissent des plans d'actions, ils sont obligés de mener au bout leurs exécutions pour produire un résultat partiel. L'ensemble de tous les résultats partiels combinés forme un résultat global et ainsi un produit final de la collaboration.

En résumé notre SMA-C bénéficie des propriétés communes des deux domaines ; à savoir, la communication et la coordination. Ainsi que des propriétés spécifiques à l'un ou à l'autre ; à savoir, l'interaction, l'intelligence sociale et la production d'un produit fini.
Dans le chapitre suivant nous exposerons la conception, l'implémentation ainsi que l'évaluation du SMA-C.

Chapitre 3

Conception, implémentation et évaluation du SMA-C

3.1 Introduction

Dans ce chapitre nous allons présenter la conception, l'implémentation ainsi que l'évaluation du SMA-C. Il s'agit d'utiliser le modèle d'agent collaborateur proposé dans le second chapitre pour concevoir, implémenter et évaluer l'architecture logicielle du SMA-C. Dans la première partie de ce chapitre nous présentons les deux outils choisis pour la conception et l'implémentation de l'architecture logicielle du SMA-C. Dans la seconde partie de ce chapitre nous abordons la conception et l'implémentation de l'architecture logicielle du SMA-C. Dans la dernière partie nous présentons l'évaluation du SMA-C en proposant également un simulateur développé à l'occasion de cette évaluation.

3.2 Outils utilisés

3.2.1 UML

Avant de présenter l'architecture logicielle, nous commençons par présenter l'outil de conception UML qui nous a servi pour représenter nos agents.
UML (Unified Modeling Language) est né de la fusion des trois méthodes qui ont le plus influencé la modélisation objet au milieu des années 90 : OMT, Booch et OOSE. Issu d'un travail d'experts reconnus, UML est le résultat d'un large consensus. UML comble une lacune importante des technologies objet, en fait il permet d'exprimer et d'élaborer des modèles objet, indépendamment de tout langage de programmation, il a été pensé pour servir de support à une analyse basée sur les concepts objet. C'est

un langage formel, défini par un métamodèle, il permet de représenter un système selon différentes vues complémentaires, à savoir, les diagrammes. Un diagramme UML est une représentation graphique, qui s'intéresse à un aspect précis du modèle ; c'est une perspective du modèle, chaque type de diagramme UML possède une structure (les types des éléments de modélisation qui le composent sont prédéfinis) et véhicule une sémantique précise (il offre toujours la même vue d'un système). Combinés, les différents types de diagrammes UML offrent une vue complète des aspects statiques et dynamiques d'un système. Ce langage opte en effet pour l'élaboration des modèles, plutôt que pour une approche qui impose une barrière stricte entre analyse et conception. Les modèles d'analyse et de conception ne diffèrent que par leur niveau de détail, il n'y a pas de différence dans les concepts utilisés. Il n'introduit pas d'éléments de modélisation propres à une activité (analyse, conception...) ; le langage reste le même à tous les niveaux d'abstraction. Cette approche simplificatrice facilite le passage entre les niveaux d'abstraction et l'élaboration encourage une approche non linéaire, les "retours en arrière" entre niveaux d'abstraction différents sont facilités et la traçabilité entre modèles de niveaux différents est assurée par l'unicité du langage.
Dans ce qui suit, nous présentons quelques notations du langage UML2, qui nous ont servis pour la modélisation de nos diagrammes. En fait, UML possède différents types de modèles, dans cette partie nous ne présentons que quelques notations simplifés du diagramme de classes (figure 3.1) et de celui de séquence (figure 3.2), que nous utilisons dans ce manuscrit.

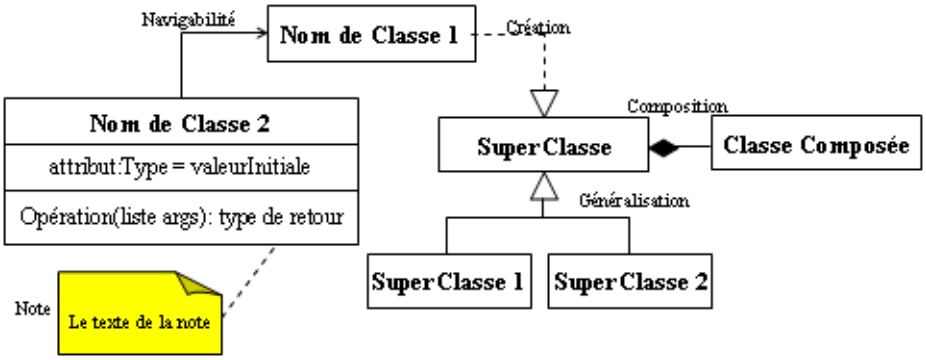

Figure 3.1: Un diagramme de classes simplifié.

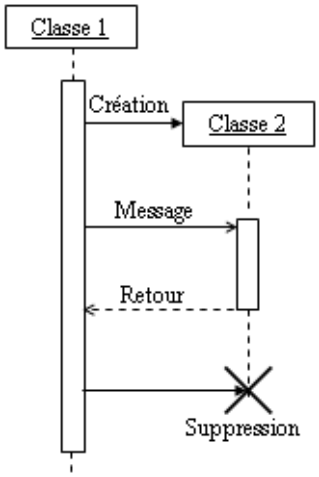

Figure 3.2: Un diagramme de séquence simplifié.

3.2.2 La plateforme JADE

JADE, pour Java Agent DEvelopment framework, est un middleware écrit en Java et se conformant aux spécifications de la Fipa. Cet environnement simplifie le développement d'agent en fournissant les services de base définis par la Fipa, ainsi qu'un ensemble d'outils pour le déploiement. La plateforme JADE peut être répartie sur un ensemble de machines et configurée à distance. La configuration du système peut être modifié dynamiquement, grâce à un mécanisme de migration d'agent au sein de la même plateforme. La plateforme JADE contient :

- Un environnement d'exécution, où les agents JADE peuvent évoluer, il doit être actif sur un hôte donné et ainsi un ou plusieurs agents peuvent être exécutés sur cet hôte.

- Une libraire de classes que les programmeurs utilisent pour développer leurs agents.

- Une suite d'outils graphiques qui permettent l'administration et la supervision des activités des agents en exécution.

Chaque instance de l'environnement d'exécution de JADE est appelée Conteneur (Container) car il peut contenir plusieurs agents. L'ensemble des conteneurs actifs est appelé plateforme. Un seul conteneur principal (main container) spécial doit être actif dans une plateforme et tous les autres conteneurs s'enregistrent dedans quand ils commencent à s'exécuter. La

figure 3.3 représente une capture d'écran de l'interface graphique de JADE présentant un plateforme JADE contenant plusieurs agents (la fenêtre de gauche) et un message ACL (la fenêtre de droite).

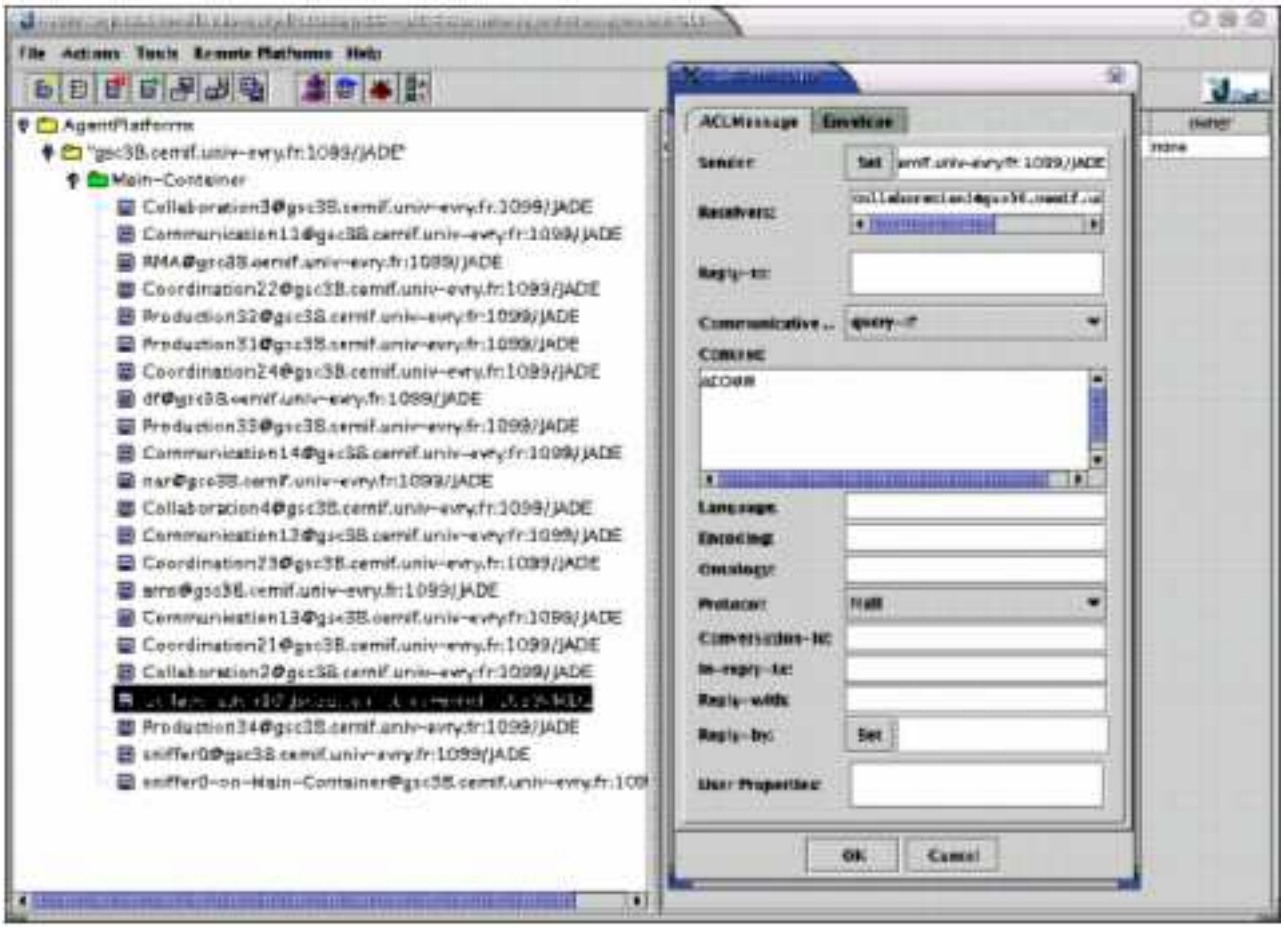

Figure 3.3: Une capture d'écran de l'interface graphique de JADE.

Le conteneur principal contient deux agents spéciaux :

- L'AMS (Agent Management System) qui fournit un service de nommage (i.e. il assure que chaque agent dans la plateforme possède un nom unique) et il représente l'autorité dans la plateforme (c'est possible de créer ou de tuer des agents dans des conteneurs distants en le demandant à l'AMS).

- Le DF (Directory Facilitator), quant à lui, il offre le service de Pages Jaunes au moyen duquel un agent peut trouver d'autres agents fournissant les services dont il a besoin dans le but d'atteindre son objectif.

JADE définit un modèle d'agent générique qui peut réaliser tout type d'architecture d'agents, allant des réactifs au BDI (Belief, Desire, Intention). Cette plateforme intègre totalement le modèle de communication de FIPA: protocoles d'interaction, enveloppe, ACL (Agent Communication Language), langages de contenu, structures de représentations, ontologies et protocoles de transport.

3.2.2.1 Les comportements

Le travail qu'un agent doit faire est effectué dans des comportements (bahaviours). Un comportement représente une tâche qu'un agent peut effectuer et il est implémenté comme un objet. Nous pouvons distinguer trois types de comportements :

- Le comportement "One-shot" qui se termine immédiatement et qui s'exécute une seule fois.
- Le comportement "Cyclic" qui ne se termine jamais et qui s'exécute chaque fois qu'il est appellé.
- Le comportement générique qui intègre un statut et qui exécute de différentes opérations dépendant de ce statut.

3.2.2.2 La communication

Le paradigme de la communication adopté est un passage asynchrone de messages. Les messages échangés par les agents JADE ont un format spécifié par le langage ACL défini par la FIPA. Ce format comprend un certain nombre de champs et particulièrement :

- L'expéditeur de message.
- La liste des receveurs.
- L'intention de la communication (performative) qui indique le but de l'envoi de ce message.
- Le contenu du message.
- Le langage du contenu i.e. la syntaxe utilisée pour exprimer le contenu, comprise par l'expéditeur et le receveur.
- L'ontologie i.e. le vocabulaire des symboles utilisés dans le contenu et leurs significations, compris par l'expéditeur et le receveur.

3.2.2.3 Les protocoles d'interaction entre agents

La FIPA spécifie un ensemble de protocoles d'interaction standard, qui peuvent être utilisés comme modèles pour construire des conversations d'agents. Pour chaque conversation, JADE distingue le rôle initiateur (l'agent sui commence la conversation) et rôle répondeur (l'agent qui s'engage dans une conversation après avoir été contacté par un autre agent). JADE fournit des comportements prêts à utiliser pour les deux rôles dans les conversations suivant les protocoles d'interaction de la FIPA. Les différents protocoles d'interaction sont : AchieveREInitiator/Responder, ContractNetInitiator/Responder, ProposeInitiator/Responder, SimpleAchiveREInitiator/Responder, SubscriptionInitiator/Responder.

3.3 Conception et implémentation du SMA-C

Dans cette partie nous présentons l'architecture conceptuelle du Système Multi-Agent pour la Collaboration (SMA-C). Nous allons construire un système de simulation multi-agent dans le but de valider le modèle d'agent collaborateur présenté dans le chapitre précédent. Ce simulateur est composé de différents agents collaborateur. Les interactions et la communication entre ces agents sont basées sur le standard de la FIPA. Ce système ne gère pas les utilisateurs humains mais tout simplement il simule la collaboration entre différents utilisateurs.
Initialement un agent serveur, que nous appelons agent générateur, crée les différents agents collaborateur. Pour chaque agent collaborateur; un agent collaboration, un agent communication, un agent coordination et un agent production sont créés. Une fois les quatre agents créés, l'agent générateur envoie les adresses de chacun d'entre eux à l'agent collaboration. La figure 3.4 présente un exemple lorsque deux clients éventuels sont connectés. Dans ce qui suit nous appelons l'agent générateur Serveur.
Le SMA-C présenté ici est principalement composé d'agents, héritant d'une classe `AgentGenerique` (figure 3.5) qui fournit les méthodes de base pour l'enregistrement des agents et leur communication. Les autres classes servent à exploiter le SMA-C en fournissant une interface et de quoi suivre son fonctionnement.

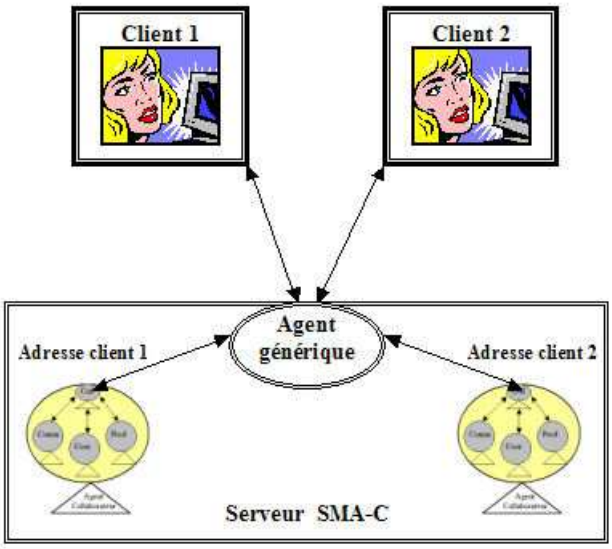

Figure 3.4: Exemple de deux agents collaborateurs générés lorsque deux clients sont connectés

Les classes d'agents `AgentCollaboration`, `AgentCommunication`, `AgentCoordination` et `AgentProduction` possèdent des classes internes qui héritent de divers comportements et protocoles d'interaction offerts par JADE : `CyclicBehaviour`, `AchieveREInitiator` / `AchieveREResponder` et `ContractNetInitiator` / `ContractNetResponder`. Nous les avons présentés dans la section 3.2.2 et ils sont détaillés dans les spécifications FIPA mais nous pouvons résumer ci-dessous leurs rôles dans notre système :

- Le comportement *Cyclic Behaviour* : écoute de messages indépendants les uns des autres échangés par les différents agents de notre système.
- Le protocle d'interaction *Rational Effect* (*RE*) : dialogue entre agents sous forme d'une proposition et d'une réponse (positive ou négative).
- Le protocole *Contract Net* : dialogue entre agents sous forme d'un appel à enchères suivi de propositions (et de réponses de l'initiateur suivant les propositions).

La classe `Serveur` est elle-même un agent, qui génère les autres agents et lance l'interface graphique. La classe `Serveur` est représentée en figure 3.6.

Ces classes sont liées entre elles de la façon représentée dans le diagramme de classes en figure 3.7.

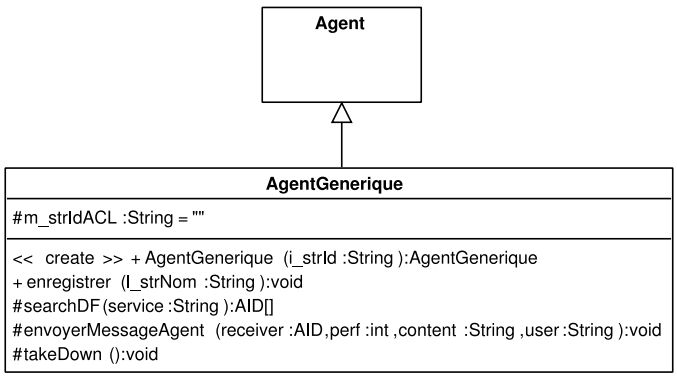

Figure 3.5: La classe `AgentGenerique`

3.3.1 Protocole de communication

3.3.1.1 Communication

La communication est décomposée de deux étapes : la création des groupes de travail et le choix de la mission effectuée par le groupe. Elle est initiée par le générateur, qui, une fois tous les agents collaborateurs créés, chacun étant identifié par un numéro unique, notifie chacun d'entre eux du démarrage de la communication. Les agents collaboration s'enregistrent auprès du Directory Facilitator[1] sous le service "collaboration".
Le protocole choisi pour la composition des groupes est le suivant : à un moment aléatoire suivant le démarrage de la communication, chaque agent collaboration i n'appartenant pas à un groupe envoie aux autres agents enregistrés comme "collaboration" une requête (`ACLMessage.REQUEST`) de composition de groupe (protocole d'interaction `AchieveRE`). Les agents collaboration acceptent ou non (aléatoirement) de le rejoindre. Chaque agent collaboration ayant accepté la requête de l'agent i rejoint le groupe i et modifie son service par "coll$i$" puis envoie le numéro de groupe à ses 3 sous-agents ainsi que la liste des missions à son agent communication. L'agent collaboration initiateur i fait de même après réception de tous les accusés de réception, pour peu qu'il ne soit pas seul.
Le protocole pour le choix de mission est similaire. Après réception de sa liste de missions, l'agent communication maître (celui de l'agent collaborateur dont l'agent collaboration i a formé le groupe) choisit aléatoirement

[1] Agent du runtime JADE servant de gestionnaire de "pages jaunes" auprès duquel les agents peuvent se référencer en tant que fournisseurs de services.

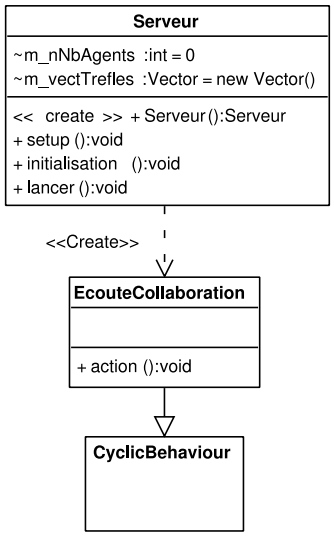

Figure 3.6: La classe `Serveur` et ses comportements.

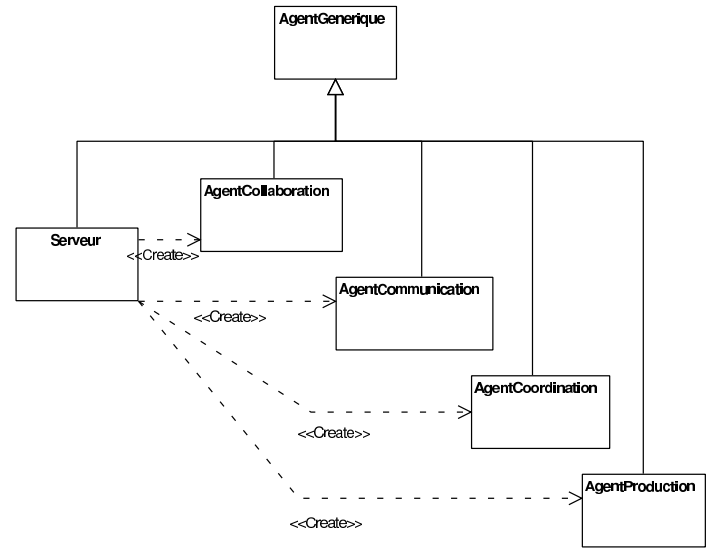

Figure 3.7: Diagramme de classes

une mission dans la liste disponible. Il l'envoie à tous les agents communication du groupe (enregistrés auprès du DF comme tels) sous le protocole `AchieveRE`. Chacun répond aléatoirement `ACCEPT` ou `REFUSE`. Si une majorité d'agents accepte, la mission est acceptée, sinon le dialogue est réitéré jusqu'à acceptation. Une fois la mission acceptée, l'agent collaboration maître est notifié et envoie ce choix à tous les autres agents collaboration du groupe, puis passe en coordination.

3.3.1.2 Coordination

La coordination débute quand un agent collaboration reçoit la mission choisie. Il envoie alors la liste d'actions correspondantes à son agent coordination. Pour chaque action, l'agent coordination maître lance une enchère, sous la forme d'un protocole `ContractNet`. L'action est envoyée à chaque agent coordination du groupe, qui répond par une offre aléatoire (flottant). A chaque réponse reçue deux cas se proposent :

- L'offre est meilleure que la meilleure jusque là : l'agent qui avait fait la précédente meilleure offre reçoit un `REJECT-PROPOSAL`.
- L'offre est moins bonne : l'agent qui a fait cette offre reçoit un `REJECT-PROPOSAL`.

Une fois toutes les réponses reçues, l'agent qui a fait la meilleure offre reçoit un `ACCEPT-PROPOSAL`. Il notifie alors son agent collaboration de l'action qu'il devra effectuer.
Une fois toutes les actions réparties, l'agent coordination maître lance la production (dans la figure 3.8, la mission est composée de deux actions : Action 1 et Action 2).

3.3.1.3 Production

La production est lancée lorsque l'agent coordination maître a réparti toutes les actions. Il envoie alors à son agent collaboration une requête de passer à la première action. Cet agent relaye cet ordre à tous les agents collaboration. L'agent collaboration devant faire cette action notifie son agent production de lancer la production. Une fois la production terminée, l'agent production retourne à son agent collaboration un signal de fin, qui est répercuté à tous les agents collaboration. L'agent collaboration maître le notifie à son agent coordination, qui envoie l'ordre de faire l'action suivante ou signale la fin de production.

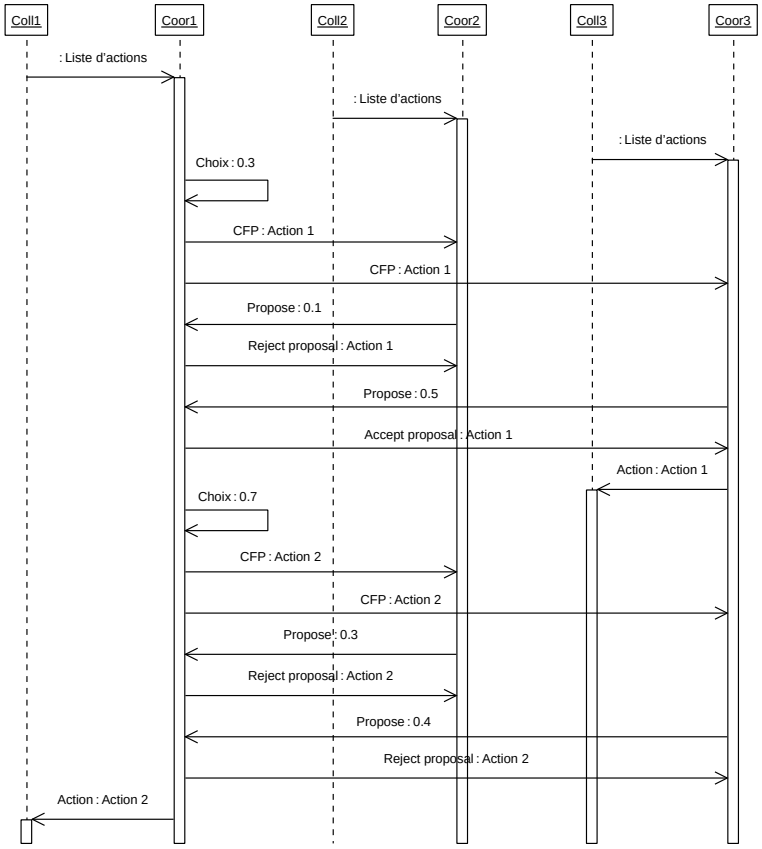

Figure 3.8: Diagramme de sequence de la répartition des actions

Le modèle peut être enrichi en ajoutant la supervision (l'observabilité des actions), qui consiste en des trames qui sont remontées de l'agent production en action à son agent collaboration, qui redistribue à tous les autres agents collaboration du groupe, qui eux-mêmes notifient leurs agents production. Dans la figure 3.9 la première action est attribuée à l'agent collaborateur 2 et la seconde à l'agent collaborateur 1, à la fin, tous les agents sont tués.

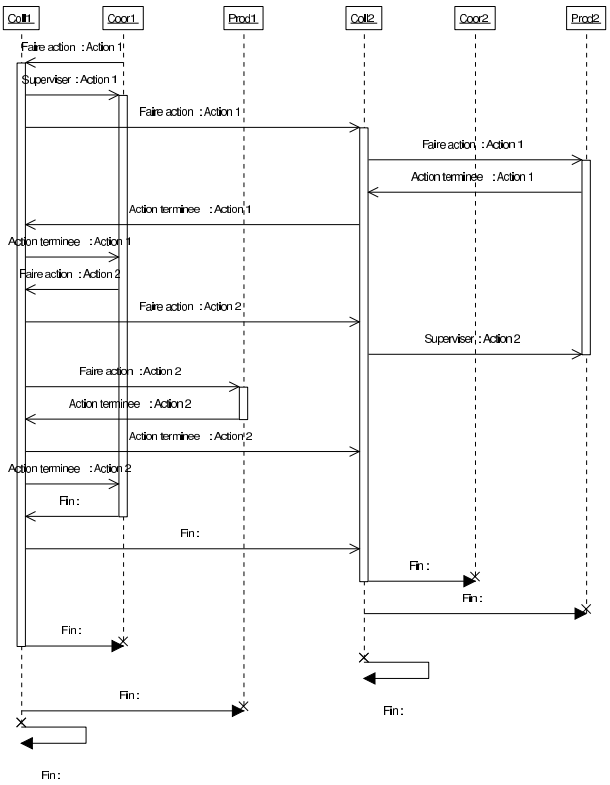

Figure 3.9: Diagramme de sequences de la production des actions

3.4 Evaluation du SMA-C

Nous commençons cette section par la présentation de l'interface graphique du simulateur développé pour évaluer les performances de notre SMA-C [KHE 05b].
Le but de cette évaluation est de valider le modèle d'agent collaborateur, présenté dans le chapitre précédent. Nous rappelons que cette simulation est effectuée sans l'implication des utilisateurs et dans les chapitres suivants, nous présentons l'intégration du SMA-C comme un noyau d'un système de téléopération collaborative qui sera évalué par des utilisateurs.

3.4.1 Interface graphique du simulateur

Dans ce qui suit nous présontons les différentes fonctionalités de l'interface graphique du simulateur, en précisant les grandeurs mesurées et observées ainsi que la classe "GuiServeur" qui regroupe toutes les fonctionalités de cette interface graphique.

3.4.1.1 Grandeurs mesurées et observées

L'exploitation du SMA se fait par l'utilisation des données suivantes :

- La sortie graphique de la production : chaque agent production en activité dessine un trait dans une fenêtre. Cette sortie permet de contrôler le bon fonctionnement global du système.

- Le décompte des messages échangés, pour une analyse fine du processus : quel émetteur, quel récepteur.

- Le nombre de messages échangés dans chaque phase : on choisit de classer ces messages par type d'agent émetteur (telle classe d'agent émet tant de messages durant telle phase).

- La mesure de la durée de chaque phase : communication, coordination, production.

Ces données peuvent être extraites pour la totalité du SMA comme pour un groupe en particulier.
La solution adoptée est de pouvoir suivre indépendamment un groupe (arbitrairement et pour plus de simplicité le premier qui se forme) comme la totalité des agents en jeu. La possibilité de suivre les échanges en temps

réel est une fonctionnalité immédiate à implémenter et qui permet de suivre facilement les différentes phases de la communication.
Les compteurs de messages émis se présentent sous la forme de tableaux à double entrée. Pour le compteur de messages échangés, on trouve horizontalement la classe d'agent émetteur (collaboration, communication, coordination ou production) et verticalement la classe d'agent récepteur (idem). Chaque message envoyé incrémente un compteur dans la case correspondante. De la même manière un tableau comportant horizontalement le type d'agent émetteur et verticalement la phase de collaboration permet de compter le nombre de messages émis dans chaque phase.
Toutes ces fonctionnalités sont regroupées par une classe `GuiServeur` qui cumule les fonctions de compteur et de sortie graphique. Une classe `FrameSortie` permet d'afficher l'état de la production (traits dessinés).

3.4.1.2 La classe `GuiServeur`

La classe `GuiServeur` (dont le diagramme de classes est présenté en figure 3.10) a deux tableaux d'entiers à double entrée `mMessagesGroupe` et `mMessageGlobal` servant de compteur de messages émis et réçus. Chaque agent doit pouvoir utiliser ce compteur, mais sans passer par des messages pour ne pas fausser les mesures. Ces tableaux sont donc membres de la classe `GuiServeur` (*static*). De la même manière, le compteur de messages émis dans chaque phase est constitué de trois tableaux d'entiers, chacun dédié à une phase : `mMessagesComm`, `mMessagesCoor`, `mMessagesProd`.
Dans le même ordre d'idée, des objets `Date` membres de la classe permettent d'enregistrer le début et la fin de chaque étape de la collaboration pour le groupe suivi. Ce groupe est identifié par le membre de classe `mGroupe`. Initialisé à -1, ce membre est mis au numéro (positif) du premier groupe qui se crée par le maître de ce dernier. Si la valeur est différente de -1 lorsqu'un groupe se crée, la valeur n'est pas changée car cela signifie qu'un groupe s'est déjà enregistré comme étant le premier.

L'enregistrement d'un message se fait avec la méthode de classe `enregistrerMessage`, qui prend en argument le numéro de groupe, le type d'agent émetteur, le type d'agent récepteur et le nombre de messages envoyés (le nombre de récepteurs). Si le numéro de groupe est le numéro enregistré comme étant celui du premier groupe, le compteur du groupe est mis à jour. Dans tous les cas, le compteur global est mis à jour. Si la fonctionnalité d'affichage temps réel est sélectionnée, à chaque enregistrement de message, l'affichage

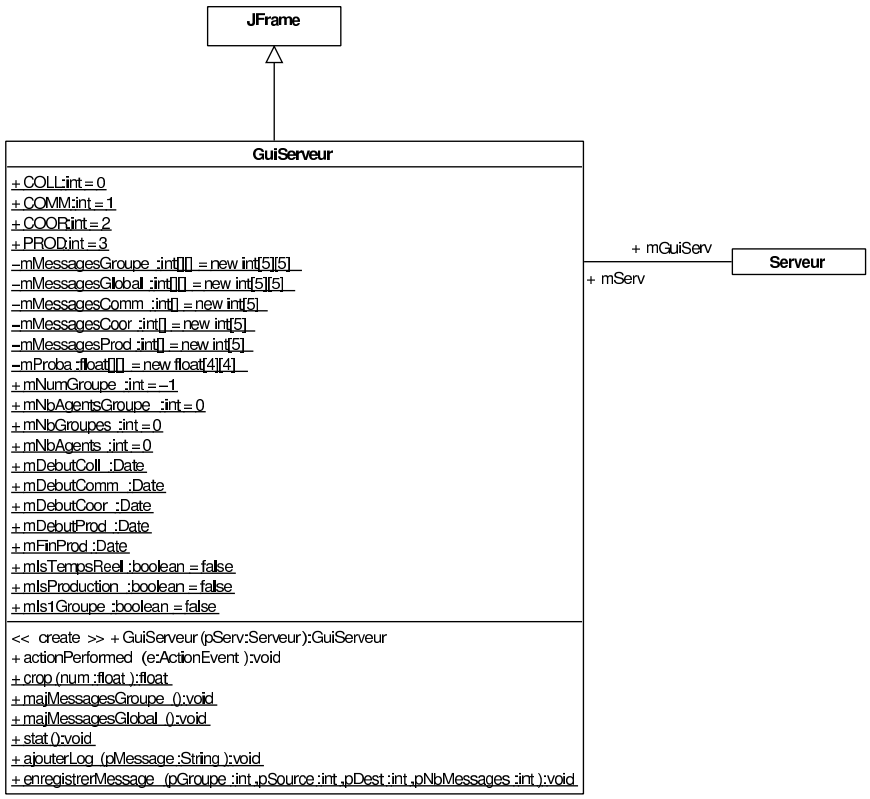

Figure 3.10: La classe `GuiServeur` comprend une partie graphique et divers compteurs.

est mis à jour.
Le principe est similaire pour l'enregistrement des débuts et fins d'étapes : l'agent qui reçoit le message signalant la fin d'une étape marque l'instant en créant un nouvel objet `Date` s'il est du groupe enregistré :

- Le début de la collaboration `mDebutColl` est enregistré par l'agent collaboration maître du groupe quand il envoie l'offre de former le groupe.

- Le début de la communication `mDebutComm` est enregistré par l'agent collaboration maître lorsque le groupe est formé.

- Le début de la coordination `mDebutCoor` est enregistré par l'agent collaboration maître lorsqu'il reçoit le choix de mission de son agent communication.

- Le début de production `mDebutProd` est enregistré par l'agent collaboration maître lorsqu'il reçoit le signal de faire la première action.
- La fin de production (et donc de collaboration) `mFinProd` est enregistrée par l'agent collaboration maître lorsqu'il reçoit le signal de fin de production (juste avant de mourir).

Enfin, le dernier point est le suivi de production. La classe `FrameSortie` (dont le diagramme de classes est présenté en figure 3.11) n'a qu'une méthode majeure, `ajouterPoint`, qui tire un trait depuis le dernier point tracé jusqu'au point passé en argument, à la couleur passée en argument. Chaque agent collaboration maître instancie une `FrameSortie`, et met à jour son contenu lorsqu'il reçoit une trame de supervision de la part de l'agent production en action du groupe. La trame de supervision contient la coordonnée du point à tracer et la couleur correspondante. Le suivi de la production implique donc l'activation de la supervision.

Les fonctionnalités de supervision et de suivi temps réel impliquent un grand nombre de trames supplémentaires échangées ainsi qu'une chute significative des performances (l'affichage graphique est rafraîchi pour chaque message). Ces fonctionnalités sont donc incompatibles avec l'évaluation de performances brutes du SMA mais permettent de suivre le bon fonctionnement de la collaboration.

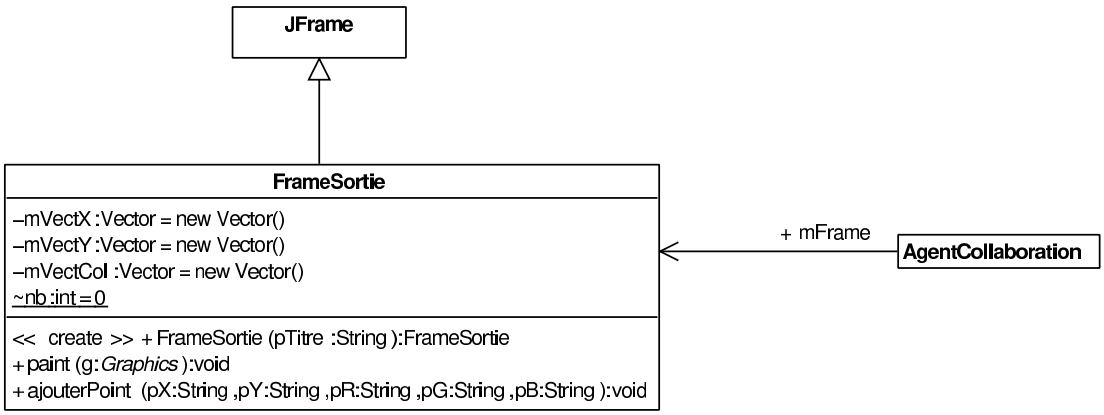

Figure 3.11: La classe `FrameSortie`.

L'interface graphique du simulateur est présentée (dans les figures 3.12, 3.13 et 3.14) à travers une simulation sur 10 agents collaborateurs avec un affichage des différentes statistiques possibles utilisées pour l'évaluation du SMA-C.
Nous présentons ci-dessous les différents types de sortie offerts par le simulateur :

- La figure 3.12 montre les statistiques du groupe : Le nombre de messages reçus et envoyés, puis le nombre d'agents du groupe, et la moyenne de messages par agent. Enfin les durées de chaque étape de la collaboration sont présentées.

- La figure 3.13 montre les statistiques globales de la collaboration : Nombre de messages reçus et envoyés, puis par agent, puis par groupe.

- La figure 3.14 montre les messages reçus et envoyés du groupe (comme la figure 3.12), puis les durées et le nombre de messages de chaque partie de la collaboration. Enfin les fenêtres graphiques de suivi en ligne de la production des différents agents dans chaque groupe crée. Dans cet exemple 6 à 8 actions sont exécutées (dessiner un segment de droite) par un groupe, une couleur et un numéro sont associé à un agent de production ayant produit le segment.

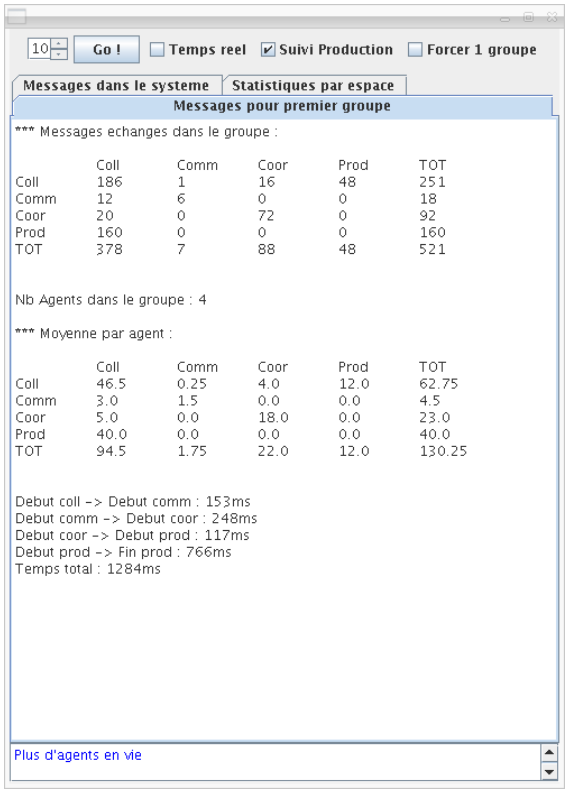

Figure 3.12: Interface graphique du simulateur : affichage des données extraites pour un groupe.

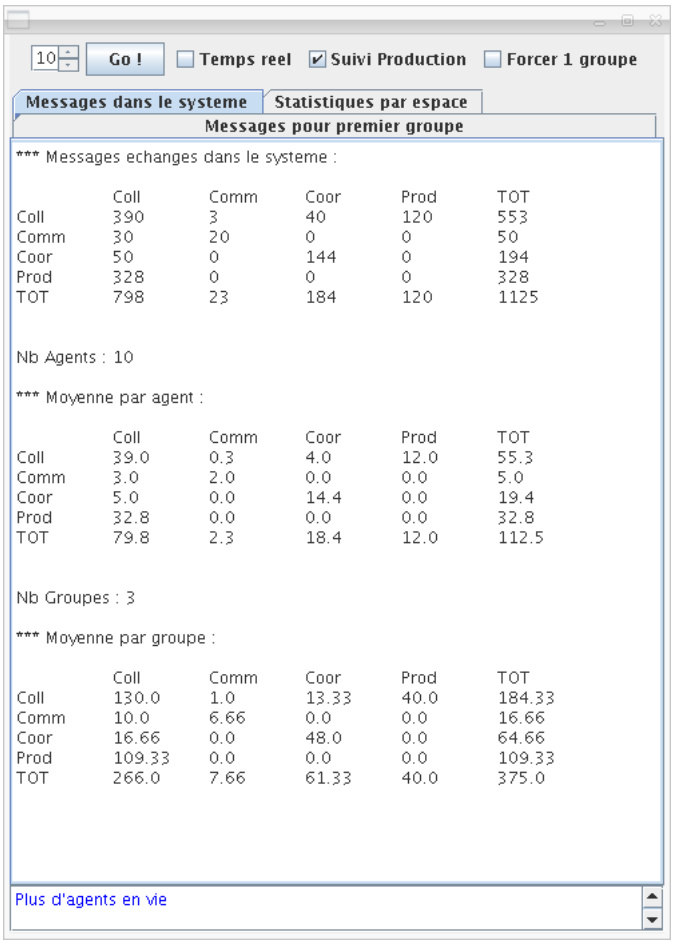

Figure 3.13: Interface graphique du simulateur : affichage des données extraites pour l'ensemble des agents.

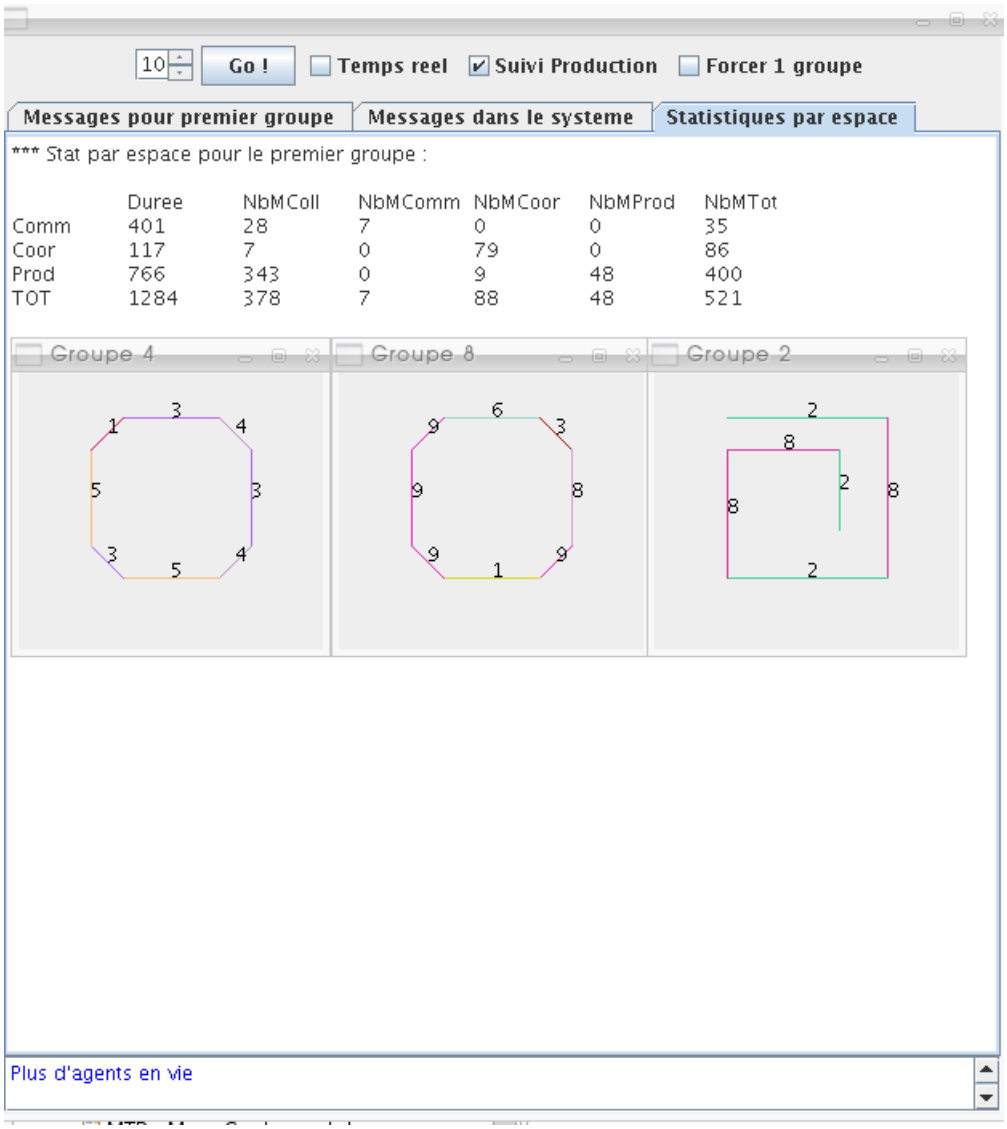

Figure 3.14: Interface graphique du simulateur : affichage du menu principal et de fenêtres de production.

3.4.2 Protocole expérimental pour l'évaluation du SMA-C

Le protocole expérimental est le suivant :

- Tous les agents joignent le premier groupe qui se forme : ceci permet d'éviter que plusieurs groupes se forment en parallèle, affectant leurs performances mutuelles. On peut ainsi lier les performances au nombre d'agents en jeu de façon directe.

- Trois missions sont proposées, chacune constituée du même nombre d'actions (8).

- Il n'y a pas de supervision (suivi de la production), la production de chaque action est instantanée. La supervision implique un grand nombre de messages, qui réduirait l'importance relative des messages de communication et de coordination, sans apporter d'intérêt majeur pour l'étude du SMA.

Les données enregistrées sont la durée de chaque étape de la collaboration et les messages émis par chaque agent durant chaque étape. L'échantillon de test est composé de 41 essais, avec des nombres d'agents collaborateurs variant de 2 à 35. Un extrait des données récupérées est donné dans les tableaux 3.1, 3.2 et 3.3.

NbAgents	Δ_t	NbMComm	NbMColl	NbMCoor	NbMProd	NbM
2	91	10	3	0	0	13
4	100	22	25	0	0	47
6	95	34	11	0	0	45
10	295	58	73	0	0	131
20	320	118	39	0	0	157
35	527	208	69	0	0	277

Tableau 3.1: Echantillon des 41 jeux de tests effectués, pour la partie "communication". NbM correspond au nombre de messages émis. Δ_t est la durée de communication, en millisecondes.

NbAgents	Δ_t	NbMComm	NbMColl	NbMCoor	NbMProd	NbM
2	126	3	0	32	0	35
4	86	7	0	80	0	87
6	122	11	0	128	0	139
10	177	19	0	224	0	243
20	361	39	0	464	0	503
35	551	69	0	824	0	893

Tableau 3.2: Echantillon des 41 jeux de tests effectués, pour la partie "coordination". NbM correspond au nombre de messages émis. Δ_t est la durée de coordination, en millisecondes.

NbAgents	Δ_t	NbMComm	NbMColl	NbMCoor	NbMProd	NbM
2	151	47	0	8	8	63
4	124	103	0	8	8	119
6	162	159	0	8	8	175
10	289	271	0	8	8	287
20	500	551	0	8	8	567
35	781	971	0	8	8	987

Tableau 3.3: Echantillon des 41 jeux de tests effectués, pour la partie "production". NbM correspond au nombre de messages émis. Δ_t est la durée de production, en millisecondes.

3.4.3 Résultats et interprétation

3.4.3.1 Communication

Comme prévu par le modèle théorique, les agents coordination et production n'émettent pas de messages pendant la communication. On note que le nombre de messages émis par les agents collaboration $NbMColl$ suit une loi affine avec le nombre d'agents en jeu $NbAgents$, avec un coefficient de corrélation de 1 (voir figure 3.15) :

$$NbMColl = 6 * NbAgents - 2 \tag{3.1}$$

Ceci correspond aux messages suivants :

- $NbAgents - 1$ messages `REQUEST` envoyés par le futur maître à ses futurs collègues
- $NbAgents - 1$ réponses
- $3 * NbAgents$ messages pour les agents communication / coordination / production pour l'envoi du numéro de groupe
- $NbAgents$ messages pour les agents communication pour l'envoi de la liste des missions

Cette première analyse confirme bien que tous les agents ont pris part au groupe et réagissent comme prévu.

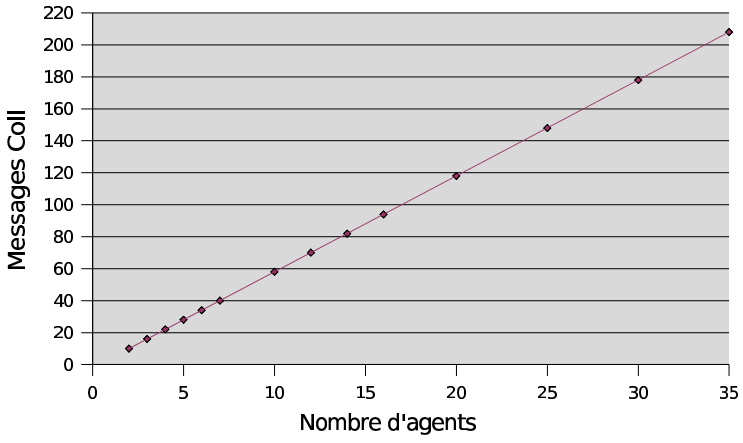

Figure 3.15: Nombre de messages émis par les agents collaboration

L'analyse des résultats du choix de mission est moins immédiate. Nous remarquons que le nombre de messages échangés ne dépend pas que du

nombre d'agents, nous pouvons le constater sur la régression linéaire sur les données qui donne un coefficient de corrélation de 0.6 (non exploitable) (voir figure 3.16). Ceci se retrouve en analysant le diagramme en camembert des proportions des messages émis par la collaboration (figures 3.17) et la communication (figure 3.18). Ces deux figures représentent les proportions de messages émis par les agents communication et les agents collaboration pendant la communication. Chaque anneau représente un test et nous remarquons que d'un essai à l'autre, les proportions relatives de chaque varient d'un multiple d'une quantité, qui correspond au nombre de messages échangés pour choisir une mission.
Nous constatons également que, d'un essai à l'autre, la part de la communication peut augmenter ou se réduire d'un certain nombre de messages : comme le choix des missions est basé sur un vote des agents, celui-ci est aléatoire et peut être reconduit plusieurs fois, augmentant par là-même le nombre de messages envoyés par les agents communication.

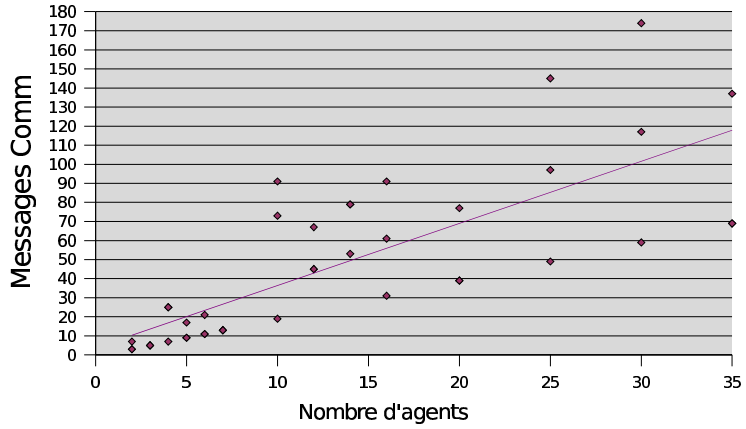

Figure 3.16: Nombre de messages émis par les agents communication

On trouve cependant que la répartition des messages entre les agents collaboration et communication est respectivement de 65% et 35%, avec une variance de 12%, due à ce facteur aléatoire.

3.4.3.2 Coordination

Là encore, nous constatons que, conformément à la théorie, seuls deux types d'agents interviennent. Les agents collaboration, qui s'envoient le choix de mission en début de coordination, et les agents coordination qui se répartissent les actions.

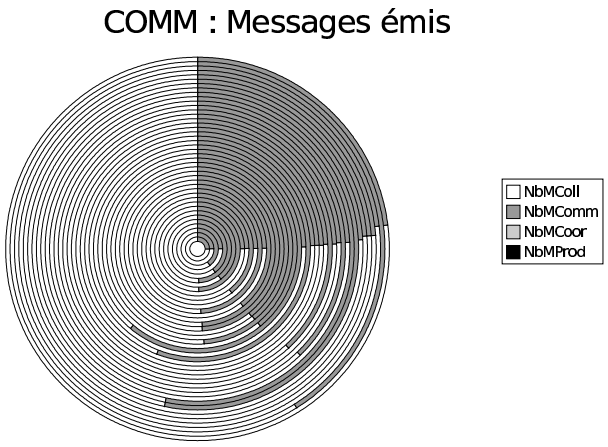

Figure 3.17: Proportions de messages émis par les agents communication et les agents collaboration

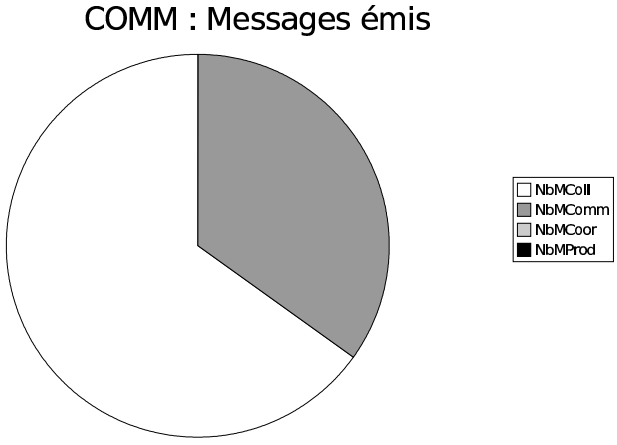

Figure 3.18: Moyennes des proportions de messages émis par les agents communication et les agents collaboration

Le nombre de messages émis par la collaboration est fixe à $2NbAgents - 1$, avec un coefficient de corrélation de 1 (figure 3.19), ce qui traduit les échanges suivants :

- $NbAgents - 1$ messages émis par le maître pour ses collègues, avec la mission choisie
- $NbAgents$ messages contenant la liste des actions, envoyés par les agents collaboration pour leurs agents coordination.

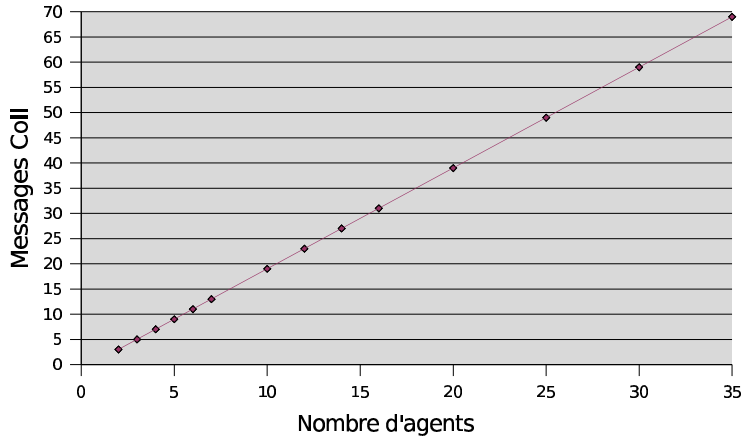

Figure 3.19: Nombre de messages émis par les agents collaboration

Une fois de plus, le nombre de messages émis par la coordination suit une relation affine du nombre d'agents, avec un coefficient de corrélation de 1 (figure 3.20) :

$$nbMCoor = 24 * NbAgents - 16 \tag{3.2}$$

Ceci s'explique par les messages suivants :

- $NbActions * (NbAgents - 1) * 3$ messages pour les enchères liées à chaque action : un `CFP`, un `PROPOSE` et un `REJECT-PROPOSAL` ou `ACCEPT-PROPOSAL` pour chaque agent du groupe
- $NbActions$ messages émis par les agents coordination remportant les enchères vers leurs agents collaboration.

Pour un nombre d'actions de 8, on trouve $24 * NbAgents - 16$.
Les messages échangés dans cette phase de la collaboration sont essentiellement des messages émis par les agents coordination (92%) : voir figure 3.21.

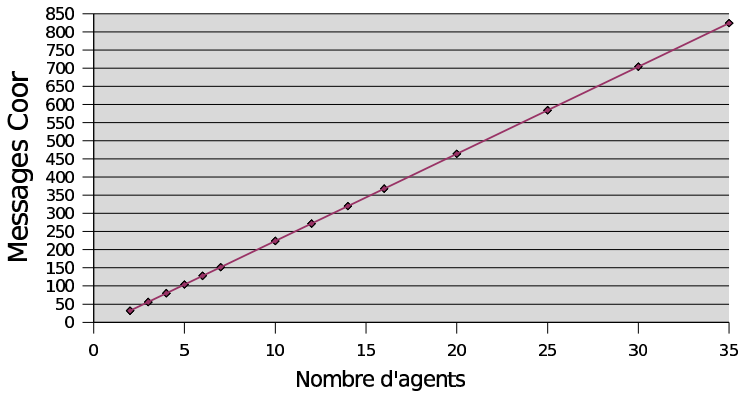

Figure 3.20: Nombre de messages émis par les agents coordination

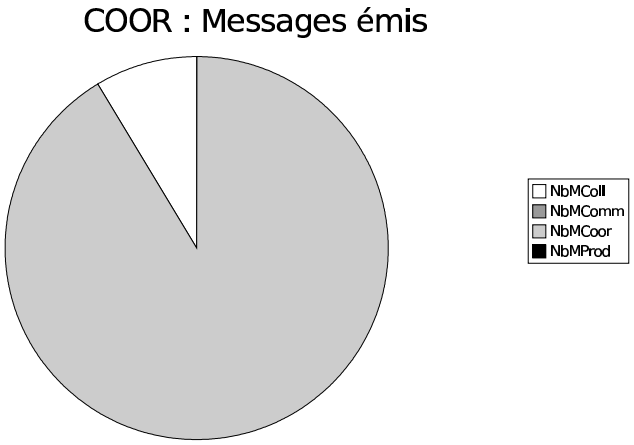

Figure 3.21: Proportions comparées des messages émis par les agents collaboration et les agents coordination.

3.4.3.3 Production

Contrairement aux résultats obtenus lors des précédentes analyses, on constate ici que certains nombres de messages ne dépendent pas du nombre d'agents : agents coordination comme production envoient un nombre de messages égal au nombre d'actions. Il s'agit respectivement des *NbActions* messages émis par l'agent coordination maître pour signaler qu'une action doit être faite et des *NbActions* messages émis par la production après accomplissement d'une action.
Du point de vue de la collaboration, on obtient encore une fois une loi affine de coefficient de corrélation 1 (figure 3.22) :

$$NbMColl = 28 * NbAgents - 9 \tag{3.3}$$

Ces messages sont répartis de la façon suivante :

- $NbActions * (NbAgents - 1)$ messages émis par l'agent collaboration maître pour signaler le début d'une action.
- $NbActions * NbAgents$ messages envoyés par l'agent collaboration à son agent production (faire ou superviser – même si ce dernier mode n'est pas exploité).
- $NbActions * (NbAgents - 1)$ messages de fin d'action.
- *NbActions* messages envoyés par l'agent collaboration maître à son agent coordination pour signaler une fin d'action.
- $NbAgents * 3$ messages pour mettre fin aux agents communication, coordination et production en fin de processus.
- $NbAgents - 1$ messages envoyés par l'agent collaboration maître pour tuer les autres agents collaboration.

Avec $NbActions = 8$, on retrouve une adéquation entre la théorie et le résultat pratique.

3.4.3.4 Aspect général

En bilan, la figure 3.23 présente les proportions moyennes des messages envoyés pendant la collaboration. Nous constatons que les agents qui ont le plus gros rôle sont les agents collaboration et les agents coordination, les premiers échangeant les information inter et intra agents collaborateurs et

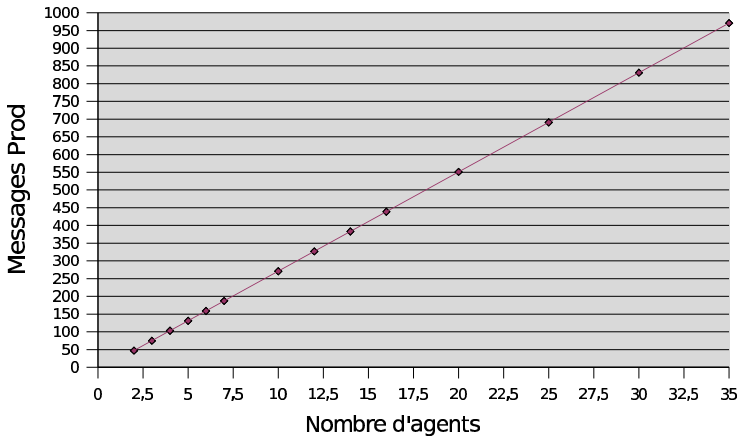

Figure 3.22: Nombre de messages émis par les agents production

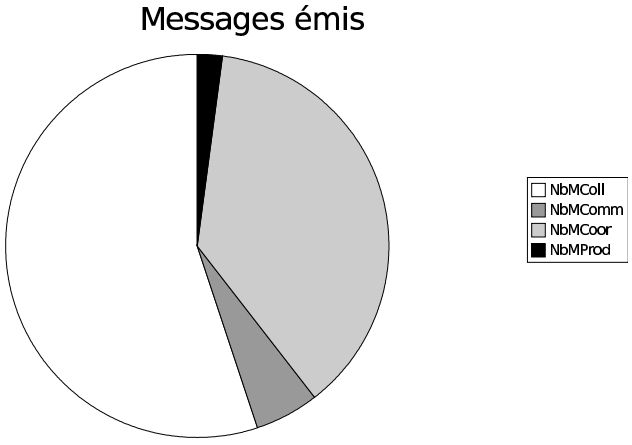

Figure 3.23: Proportions moyennes comparées des messages émis par les quatre types d'agents

les seconds s'occupant de la plus grosse négociation de la collaboration ainsi que de la bonne chronologie des tâches.
Les performances du SMA ont été mesurées en terme de relation entre le temps d'exécution et le nombre de messages échangés. Tous les tests ont été effectués sur la même machine, Pentium 4 2.4GHz/512Mo sous Linux Slackware avec noyau 2.6.11, JVM 1.4.02.
Nous trouvons une relation directe entre le temps d'exécution D_t (en millisecondes) et le nombre de messages échangés NbM, avec un coefficient de corrélation quasi-parfait de 0.99 (voir figure 3.24) :

$$D_t = 0.81 * NbM + 131.77 \tag{3.4}$$

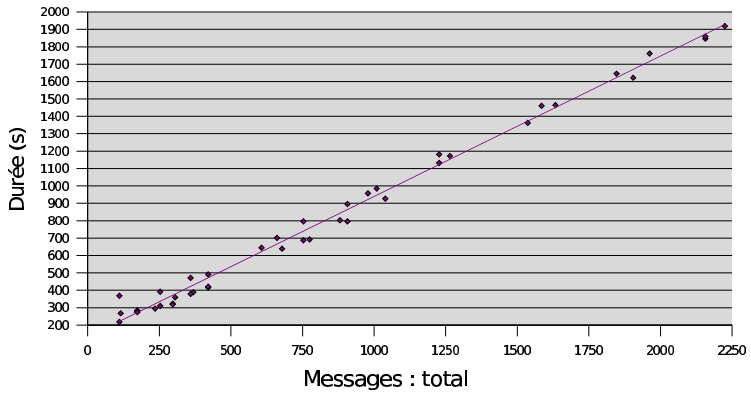

Figure 3.24: Durée d'exécution en fonction du nombre de messages émis

3.5 Bilan

Dans ce chapitre, nous avons présenté la conception, l'implémentation ainsi que l'évaluation du SMA-C.
Ensuite, nous avons présenté l'architecture d'implémentation du SMA-C, le protocole de communication des agents au sein de ce système et l'évaluation de ce dernier en utilisant le simulateur développé pour cette occasion.
Les résultats présentés nous permettent de mettre en exergue deux points majeurs : La correspondance entre le modèle théorique et les résultats pratiques est parfaite, montrant par là même que l'on peut prévoir sur le papier les lois qui régissent les nombres de messages échangés, qui seront respectées par l'implémentation (prouvant également que tous les agents fonctionnent, et comme prévu). La relation entre nombre de messages échangés et performances du système pouvant être sortie de façon claire, il est immédiat de borner le nombre maximal d'agents collaborateurs que l'on peut utiliser à niveau de performances fixé, une fois la loi régissant le nombre de messages en fonction du nombre d'agents établie.
Nous avons démontré, durant ce chapitre, que le SMA-C est un système stable pour la collaboration durant toutes ses phases : la communication, la coordination et la production, quoique la production ne dépend pas directement de notre SMA-C mais plutôt du type du système à manipuler et du travail à produire. Dans le chapitre suivant nous allons appliquer notre SMA-C dans un système de téléopération pour le rendre collaboratif. Nous allons voir son impact sur un tel système et son apport pour les utilisateurs d'un tel système de téléopération.

Chapitre 4

Application à la téléopération collaborative via Internet

4.1 Introduction

La téléopération consiste en la commande et manipulation des systèmes robotiques, à distance. Ce mode de contrôle permet d'effectuer des tâches complexes, voire impossibles pour l'homme. En effet, la téléopération permet aussi d'effectuer des interventions en milieu hostile, tels que des manipulations basiques connues et maîtrisées par l'homme. Les domaines d'application de la téléopération sont très variés et touchent la plupart des grands thèmes de recherche (recherche médicale, spatiale etc.).
En ce qui concerne les systèmes de téléopération via Internet, plusieurs projets de natures différentes ont vu le jour au cours de ces dix dernières années.

Ce chapitre a comme objectif de tester le SMA-C développé dans le chapitre précédent sur une application réelle avec des données et des contraintes réelles. Nous commençons ce chapitre par une présentation du domaine de la téléopération avec une présentation de quelques systèmes de téléopération existants via Internet. Dans la seconde partie de ce chapitre nous proposons de tester notre SMA-C sur le premier système en France de téléopération en réalité augmentée via Internet ARITI[1] (Augmented Reality Interface for Teleoperation via Internet) développé au LSC (Laboratoire des Systèmes Complexes) en 1998 et référencé sur le site web de la NASA[2] depuis janvier 2000. Il s'agit donc d'étendre le système ARITI avec de nouvelles fonctionnalités de collaboration en proposant une nouvelle version d'ARITI Collaboratif (ARITI-C) basée sur le SMA-C [KHE 05a]. Nous

[1] http://lsc.univ-evry.fr/Projets/ARITI/index.html
[2] http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html

présentons son architecture logicielle, son Interface Homme Machine (IHM) ainsi qu'une description des différentes missions et actions prises en compte pour la téléopération collaborative.

4.2 Quelques systèmes de téléopération

Dans ce qui suit, nous présentons quelques exemples de projets et/ou de systèmes, illustrant la téléopération via Internet d'une part et la téléopération collaborative d'autre part.

4.2.1 Le projet Mercury

Dans le projet Mercury [GOL 95] (figure 4.1), un robot (de type SCARA) muni d'une caméra et d'un système pneumatique, fut le premier système à permettre l'interaction avec le monde réel via Internet. Le robot offre la possibilité à l'utilisateur de visualiser les objets se trouvant dans un bac rempli de sable, au travers d'une caméra dont il contrôle la trajectoire (cette dernière étant liée à l'effecteur du robot). De plus, le projet Mercury permettait à l'opérateur d'utiliser le système pneumatique afin d'envoyer un jet d'air comprimé dans le bac, et ce, à distance. Ce système est mono-utilisateur et il n'admet pas d'autres superviseurs lorsque il est en cours d'utilisation. Nous pouvons également regretté l'absence du retour vidéo pendant le déplacement du robot.

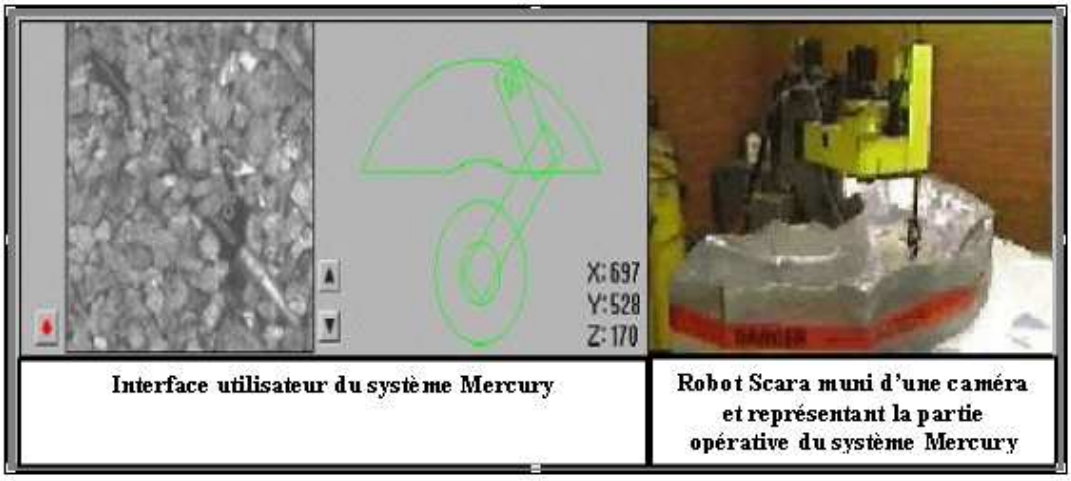

Figure 4.1: Le système Mercury

4.2.2 Australia's Telerobot on the Web

Le bras manipulateur [TAY 95] (de type ASEA IRB) permet de manipuler à distance des objets. L'utilisateur envoie la position et l'orientation de la pince axe par axe pour la placer à côté de l'objet. Il peut contrôler l'ouverture et la fermeture de la pince. Deux caméras vidéo sont utilisées pour permettre à l'utilisateur de bien placer la pince et de manipuler l'objet. Par contre il est difficile de positionner la pince et de manipuler l'objet. Ceci nécessite une grande concentration et un bon entraînement de la part de l'utilisateur. Le retour vidéo se fait à chaque exécution d'une tâche (pas de capture et de transfert d'images avant l'exécution d'une tâche). Le temps minimum de réponse du système est entre 15 et 20 secondes. Le système ne supporte pas plusieurs superviseurs, c'est-à-dire les autres utilisateurs ne peuvent pas voir ce qui se passe avec le robot tant qu'ils ne le contrôlent pas. De même, un seul utilisateur à la fois peut contrôler le bras manipulateur.

4.2.3 KhepOnTheWeb

Le robot mobile miniature, type Khepera [SAU 98] permet la navigation avec évitement d'obstacles. L'utilisateur spécifie la position et la vitesse du robot et les envoie ensuite pour l'exécution. Le système utilise deux caméras vidéo (une externe et l'autre embarquée sur le robot). L'utilisateur peut contrôler la caméra externe (orientation et zoom). Les images vidéo envoyées sont de type JPEG. Le serveur d'images utilise une technique de compression qui est supporté uniquement par le navigateur "Netscape". Un seul retour vidéo est possible à un instant donné (caméra externe ou embarquée). Le système tourne sous un système d'exploitation Windows 95 qui n'est pas stable (n'est pas fait pour ce genre d'application), ce qui oblige à redémarrer le serveur à chaque fois. Ce système est également limité par rapport aux observabilités des actions (pas d'autres superviseurs possibles).

4.2.4 Xavier

Un autre robot mobile [SIM 98] permettant aussi la navigation avec évitement d'obstacles. Le robot est autonome et capable de recevoir des commandes vocales. Une caméra embarquée sur le robot permet de visualiser ce que voit le robot. L'utilisateur choisi des tâches de haut niveau prédéfinies par le système. Le système retourne des informations de position et d'orientation toutes les 5 à 10 secondes ainsi qu'une image vidéo (de

type GIF) toutes les 20 secondes. Le système supporte un seul utilisateur à chaque fois.

4.2.5 RHINO

Un robot mobile [SCH 98] permet la navigation avec évitement d'obstacles. L'interface du système utilise la réalité virtuelle (le robot est modélisé et l'environnement est semi-modélisé). Deux caméras réelles sont utilisées, une embarquée sur le robot permettant de visualiser ce que voit le robot et une autre caméra fixe qui visualise le robot dans son environnement. Deux caméras virtuelles sont utilisées de la même façon que les réelles. Elles permettent à l'utilisateur d'observer le déplacement du robot virtuel comme résultat de l'exécution de la tâche. Ce système ne supporte qu'un seul utilisateur à chaque fois et ne possède pas de retour vidéo pendant le déplacement du robot. Une autre limitation est dû au fait que l'interface utilisateur (simulation virtuelle) est basée sur des librairies "OpenGL" qui ne sont pas supportées par défaut par les navigateurs Internet.

4.2.6 PumaPaint project

Le bras manipulateur PUMA 760 [STE 98] est capable de reproduire un dessin réalisé par un utilisateur depuis une interface de dessin dédiée (qui ressemble à celle de "paintbrush sous windows". Il existe 4 couleurs (rouge, vert, bleu et jaune) et un pinceau pour chaque couleur. Suivant la couleur et la trajectoire dessinée, le bras manipulateur va prendre le bon pinceau et reproduire la trajectoire réalisée par l'utilisateur. Après chaque tâche l'utilisateur peut visualiser le résultat en choisissant une des deux caméras dédiées pour ce retour vidéo. Pas de retour vidéo pendant la réalisation de la tâche. Le système supporte un seul utilisateur à chaque fois.

4.2.7 Le projet Ouija 2000

Le projet Ouija 2000 [GOL 00], représenté sur la figure 4.2, présente la première fois la possibilité de rajouter à la notion de téléopération, celle de travail collaboratif. Le robot, muni d'une caméra, permet aux différents utilisateurs de visualiser une planche Ouija. La planche Ouija, est une planche en bois marquée de l'alphabet ainsi que par des mots court comme OUI, NON ou encore AU REVOIR. Utilisée normalement lors de séance de spiritisme, elle est utilisée dans ce projet dans un but de téléopération collaborative.

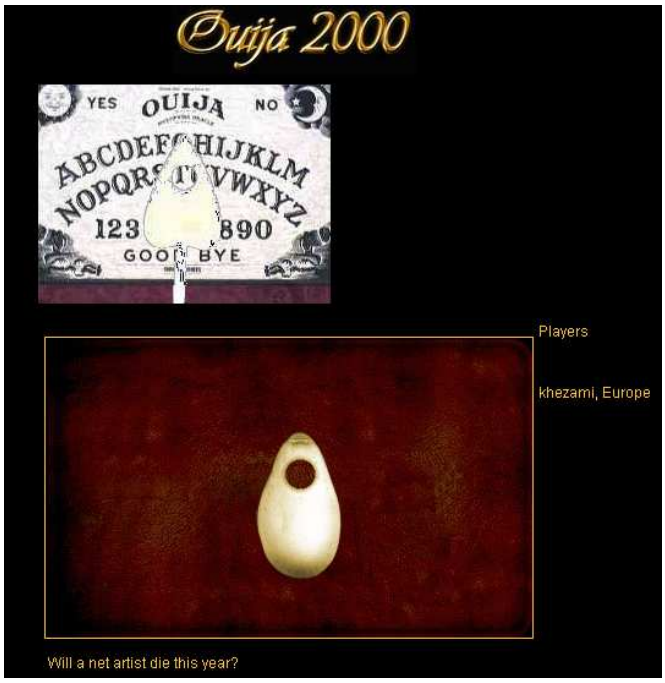

Figure 4.2: Le système Ouija

Ce projet permet donc à plusieurs utilisateurs de répondre à une question commune, en faisant bouger leurs souris vers la réponse qu'ils croient exacte. Les choix des différents utilisateurs correspondent alors, à la trajectoire de la souris. Tous les choix sont ainsi pris en compte, pour une décision commune dans le mouvement final du robot. Nous pouvons signalé des limitions dans son processus de collaboration, en effet les utilisateurs ne peuvent pas superviser les actions des autres utilisateurs présents dans le système et ils ne peuvent même pas discuter ensemble.

4.2.8 Le système Tele-Actor

Le système Tele-Actor [GOL 01], [GOL 02] développé par le même laboratoire que le projet Ouija 2000, permet de prendre en compte les décisions de plusieurs opérateurs, et ce via un système de vote (figure 4.3). La partie physique d'un tel système est alors assurée par un humain, qui se charge, en fonction de la décision finale issue du système de vote, d'effectuer la tache qui lui incombe.
Avec de tels systèmes, la téléopération collaborative et plus particulièrement la téléopération collaborative via Internet, commençait à se poser le problème de la téléopération distribuée, donc arrivait aux frontières des systèmes

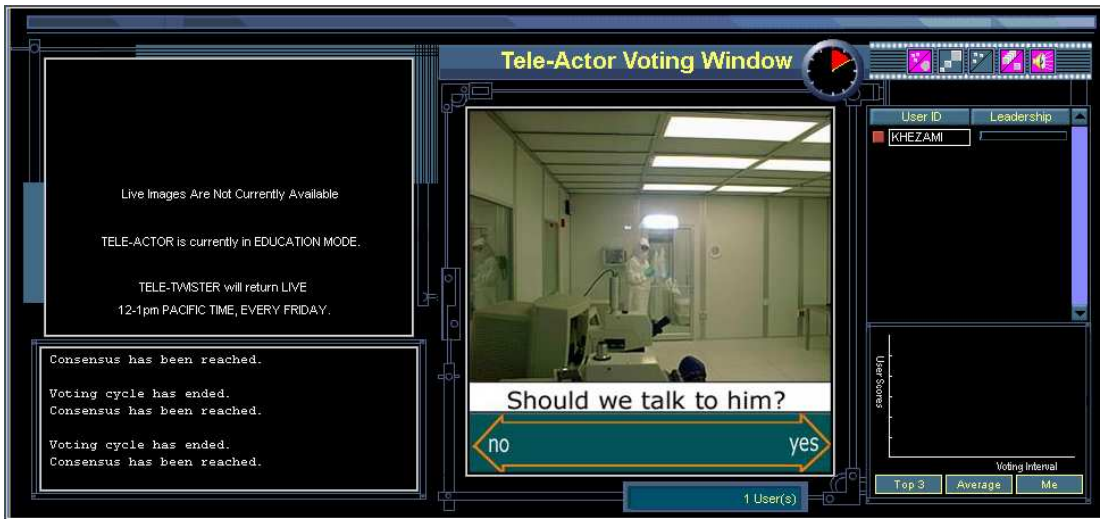

Figure 4.3: Le système Tele-Actor

multi-agents. Ce système permet à plusieurs utilisateurs de collaborer ensemble par contre ils ne peuvent pas discuter ensemble. Comme pour le système précèdent, les différents utilisateurs ne peuvent pas superviser les actions des autres utilisateurs présents dans le système.

4.2.9 E-productique

Dans le travail de [LEP 04], une architecture logicielle pour l'e-productique : SATURNE pour "Software Architecture for Teleoperation over an UnpRedictable NEtwork" a été proposée. Elle repose sur un noyau de services de communication autour duquel gravitent des modules spécifiques aux machines visées. La prise en compte des aléas du réseau est réalisée à l'aide d'un capteur logiciel affichant ses informations, sous une forme statique et dynamique à l'utilisateur, et dont les valeurs peuvent être utilisées pour une gestion de type mode de marche. Plusieurs applications ont été construites autour de SATURNE. La première machine pilotée[3] a été une caméra de type SONY EVID-31. Equipée d'un socle motorisé, elle peut effectuer des mouvements de rotation horizontale sur 200° et verticale sur 120°. Elle est de plus pourvue d'une focale variable. Dès 1999, elle a été mise en œuvre et rendue accessible sur l'internet. L'applet de contrôle contient une image statique de la scène. Elle est cliquable et permet donc l'orientation de la caméra en fonction de ce que l'utilisateur souhaite regarder. L'utilisateur a la possibilité aussi de régler le facteur de zoom et les boutons prédéfinis

[3]http ://similimi.univ-brest.fr

peuvent permettre la visualisation de lieux précis. La deuxième application consiste en la commande d'un bras manipulateur à cinq degrés de liberté de type Ericc permettant la saisie et le déplacement de petits objets. Il est connecté au serveur par une liaison série RS232 et ses mouvements sont filmés en permanence par deux caméras. La première est à focale variable placée en face du robot. La deuxième est fixe et fournit une image vue du dessus. L'utilisateur peut contrôler le robot à travers une applet de contrôle similaire à celle de la caméra SONY sauf qu'elle intègre une partie pour le contrôle du robot. Ce système ne supporte qu'un seul utilisateur à la fois.

4.2.10 Le système ARITI

Le système ARITI [OTM 00c], [OTM 00b] est un système de téléopération permettant le contrôle d'un robot, et utilisant les techniques de la réalité augmentée. ARITI[4] (Augmented Reality Interface for Teleoperation via Internet) constitue un système expérimental de télétravail via l'Internet. Au-delà de la réalisation technique, les apports d'ARITI résident dans la mise en œuvre d'assistances à l'opérateur humain pour la perception de la scène, la supervision des tâches et le contrôle du robot notamment par l'insertion de guides virtuels actifs [OTM 00a].

Les techniques utilisées ont permis d'apporter à l'opérateur en situation de télétravail, une assistance à la perception de l'environnement et à la commande d'un robot (en mettant en œuvre le concept de guides virtuels). Ces assistances ont pour objectif l'amélioration de la précision et de la sécurité du déroulement de la tâche. Le système ARITI est développé sous l'environnement Linux. Il est divisé en trois parties : le site distant (ou le site esclave), le site opérateur (ou le site maître) et la partie communication qui relie les deux sites ; qui peuvent être résumés comme suit : Le site esclave contient un robot, une caméra et un module de communication serveur (implémenté en C++) partagé par tout opérateur connecté au système. Le site maître contient un opérateur et une machine connectée à Internet (disposant d'un navigateur Internet ou bien un interpréteur Java installé). Un module IHM qui gère les interactions entre l'opérateur humain et le module de communication client. Ce module permet à l'opérateur de spécifier le mode de travail désiré à travers ces trois modules : Téléopération, Téléprogrammation ou Télésupervision. Chacun des trois modules utilise un module central appelé module Réalité Mixée qui combine les

[4] http://lsc.univ-evry.fr/Projets/ARITI/index.html

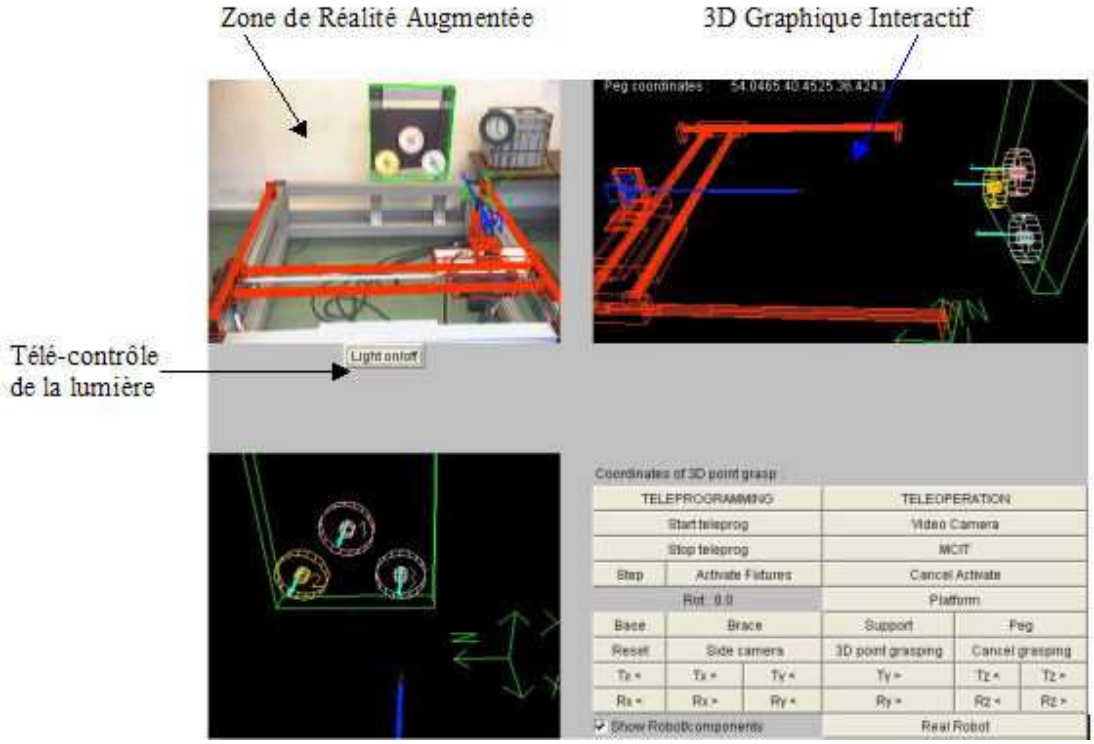

Figure 4.4: L'interface du système ARITI

deux mondes virtuel et réel dans le but de fournir une IHM conviviale. Le module Réalité Mixée joue aussi le rôle l'intermédiaire entre les différents modes de travail et le module de communication client. Il est à noter que dans ce système un seul utilisateur à la fois peut commander le robot réel. Par contre tous les utilisateurs connectés à ARITI peuvent superviser les actions de l'utilisateur qui a la main mais ils ne peuvent pas discuter ensemble.

La figure 4.4 présente l'interface utilisateur de ce système. L'interface contient quatre zones. Dans la zone de réalité augmentée (la zone en haut à gauche), un robot virtuelle est superposé sur le robot réel pour permettre à l'utilisateur de commander d'abord le robot virtuel et après validation, il va contrôler le robot réel. A l'aide du tableau de bord (la zone en bas à droite), l'utilisateur peut choisir son mode de contrôle (téléprogrammation ou téléopération), d'activer ou pas les guides virtuels ... Les deux dernières zones (la zone en haut à droite et celle en bas à gauche) permettent de donner à l'utilisateur d'autres points de vue du robot virtuel pour lui faciliter la manipulation. Enfin, le bouton de télécontrôle de lumière permet à l'utilisateur d'allumer ou d'éteindre la lumière sur le site esclave.

L'étude de l'état de l'art que nous avons effectué, nous a permis de remarquer la constante amélioration des systèmes de téléoperation. Nous avons également constaté la croissance du nombre de système téléopérés via Internet. Ceci a créé un nouveau besoin qui est celui de faire travailler plusieurs personnes distantes sur un même site, et depuis nous assistons à l'émergence de quelques systèmes dits multi-utilisateurs. Grâce aux efforts menés en intelligence artificielle pour modéliser le travail collaboratif et grâce aussi à l'amélioration du réseau et sa démocratisation, ce type de système commence à voir le jour.

4.3 Architecture logicielle de ARITI-C

Il s'agit d'utiliser le SMA-C développé dans le chapitre précédent pour proposer un collecticiel permettant un travail collaboratif avec le système ARITI. Ce collecticiel consiste à rendre possible le travail en commun de plusieurs personnes distantes sur une même mission effectuée par un même robot, la bonne coordination des tâches étant assurée par le SMA-C. La figure 4.5 illustre la transformation du système mono-utilisateur ARITI en un système multi-utilisateur collaboratif ARITI-C.
Chaque utilisateur qui se connecte à ARITI-C via Internet peut collaborer avec d'autres utilisateurs connectés de part le monde pour contrôler le robot réel. Il peut, également, discuter avec eux ou tout simplement les superviser. Un utilisateur peut, s'il le souhaite, contrôler la lumière.
Dans ce qui suit nous appelons "utilisateur" l'acteur humain et "client" le client d'un serveur informatique.
Nous allons maintenant présenter l'architecture logicielle de ARITI-C. Les développements ont été réalisés en Java, en utilisant la plateforme JADE.

4.3.1 Architecture générale

Faisant intervenir différents matériels en parallèle (robot, caméra vidéo, boitier de contrôle de la lumière), l'ancienne version du système ARITI est constituée de trois serveurs et de trois clients. Il s'agit des clients/serveurs vidéo, commande (pour contrôler le robot réel) et lumière pour allumer et eteindre la lumière. La figure 4.6 présente l'architecture client/serveur

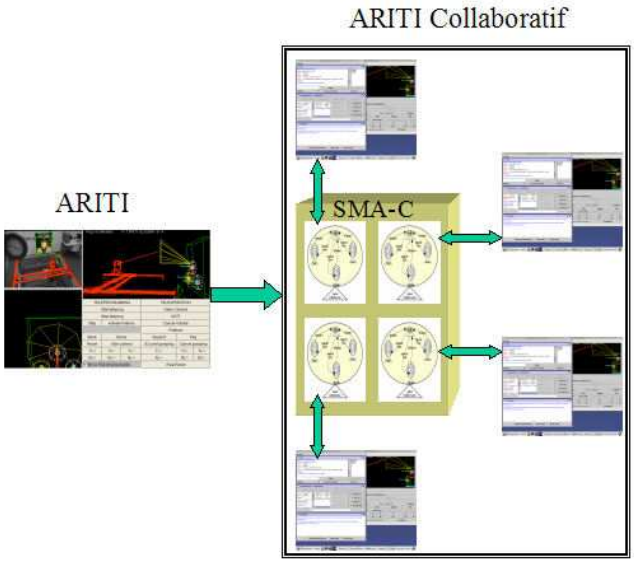

Figure 4.5: Illustration de la transformation de ARITI en ARITI-C

globale du système ARITI.

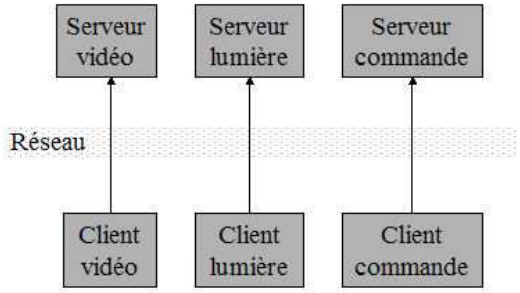

Figure 4.6: Architecture client-serveur simplifiée du système ARITI

Par simplicité de conception et d'implémentation, le serveur SMA-C est un serveur supplémentaire ajouté au système. En outre, cet ajout permet de garder la compatibilité avec les anciens clients.
Dans le système ARITI-C, contrairement au modèle idéal testé dans le chapitre précédent, c'est l'utilisateur et non le SMA qui est amené à faire des choix. Pour cela, chaque agent communication, coordination et production communique directement avec un client via le réseau. En outre, un agent générateur (comme son nom l'indique, c'est un agent qui génère d'autre agents) crée les agents collaborateurs à la demande et, de par son rôle de chef

d'orchestre, c'est lui l'agent idéal pour gérer le fonctionnement global du système : enregistrement d'événements, vérification d'identité, etc. Comme il ne peut y avoir qu'un seul client pour la commande du robot à un instant donné, c'est également cet agent générateur, unique, qui va jouer ce rôle de client commande, centralisant et répercutant les commandes envoyées par les divers utilisateurs. Par abus de langage, on pourra nommer l'agent générateur "serveur".
Enfin, un Système de Gestion de Bases de Données (SGBD) récupère les informations diverses recueillies lors de la collaboration. Son client est donc naturellement l'agent générateur.

Pour résumer, les clients d'ARITI-C sont des clients réseau pour :

- L'agent générateur, qui les identifie, leur fournit des informations diverses et crée leurs agents collaborateurs quand le besoin est soulevé.
- L'agent communication, qui sert de relai de discussion (*chat*) au sein d'un groupe. Il sert également à choisir une mission. Le choix de la mission fait partie de la communication car c'est l'utilisateur maître qui choisit la mission, elle sera ensuite transmise aux autres utilisateurs. Donc, durant le choix de la mission il n'y a aucune coordination entre les utilisateurs et c'est pour cela que cette tâche fait partie de la communication et non pas de la coordination.
- L'agent coordination, qui sert à répartir les actions pour une mission choisie.
- L'agent production, qui concerne l'utilisation du robot réel pour l'utilisateur en manipulation, et le suivi des manipulations (supervision) pour les autres.
- Le serveur vidéo.
- Le serveur lumière.

L'architecture résultante est décrite en figure 4.7. Pour des raisons de lisibilité de la figure, les autres agents (agent collaboration –qui ne communique pas directement avec le client– et agents pour les autres clients) ne sont pas représentés dans le runtime de JADE.
Dans ce qui suit, nous résumons la façon dont sont coordonnés tous les éléments d'ARITI-C. Pour référence, la figure 4.8 et sa légende en figure 4.9 illustrent cette section.

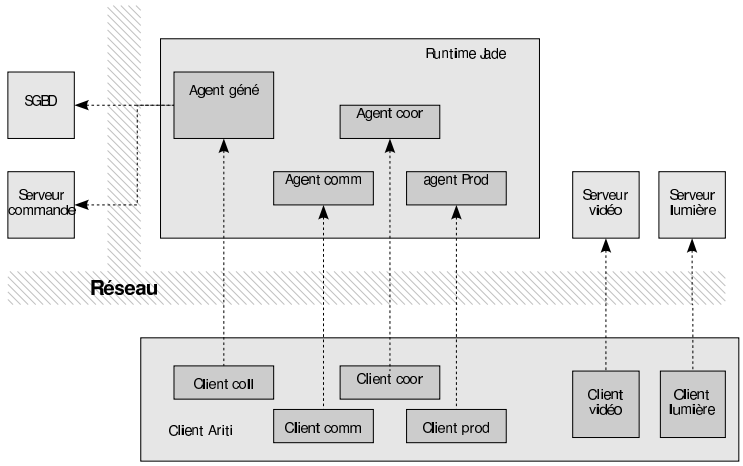

Figure 4.7: Architecture client-serveur simplifiée du système ARITI-C

Le système de ARITI-C est essentiellement constitué d'une interface englobant divers clients ainsi que de multiples serveurs dont le SMA. Afin de communiquer avec le SMA-C, un client (Coll) communique avec l'agent générateur (Gene). Ce dernier crée les agents collaborateurs (quadruplets agent collaboration, agent communication, agent coordination, agent production), qui communiquent chacun avec trois clients (client communication, client coordination et client production car l'agent collaboration ne communique pas directement avec le client collaboration) , et qui communiquent entre eux au sein du SMA. Un agent collaborateur est associé à un et un seul utilisateur du groupe.
Nous rappelons que l'agent collaborateur est celui qui collabore avec les autres agents collaborateurs et que c'est un agent abstrait, composé de quatre agents implémentés.
L'agent générateur (serveur) communique avec la commande du robot et également avec le SGBD afin d'enregistrer les informations de la collaboration et de fournir les missions et leur contenu aux agents collaborateurs.
Les autres éléments qui gravitent autour de ce système sont plus simples. Un client est dédié à chacun de ces serveurs : lumière, vidéo et enregistrement des guides. Ce dernier serveur enregistre des fichiers de sauvegarde des guides, qui sont récupérés par http par l'interface client. Cette interface récupère de la même manière le fichier des paramètres de calibration de la caméra. Ce fichier peut être généré de deux manières : soit via l'interface d'administration php, qui génère un fichier avec des paramètres prédéfinis (solution de secours), soit via le script `calibration.php`, qui

génère le fichier en fonction d'une requête http que le client lui envoie.

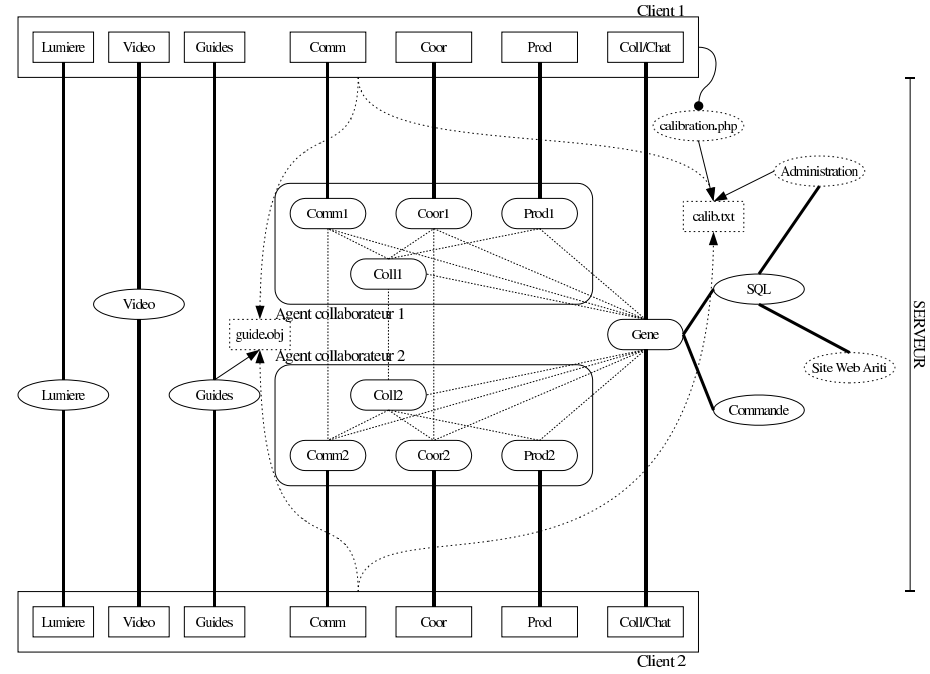

Figure 4.8: Architecture générale de ARITI-C

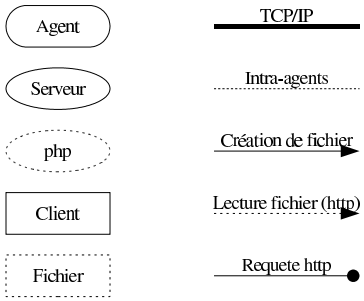

Figure 4.9: Légende pour la figure 4.8

Enfin, l'interface web d'administration comme le site web d'ARITI-C communiquent directement avec le SGBD, mais seule l'interface d'administration peut en modifier le contenu. La structure de la base de données est présentée dans la partie 5.4.1.

4.3.2 Client collaboration

Pour des raisons techniques, l'interface client cache en fait un client pour le chat et trois clients, un pour chaque agent (chaque client et donc chaque type d'agent communique sur un port différent). Le client appelé `ClientColl` (pour Client Collaboration) a le rôle de client de chat mais aussi de client pour l'ensemble des messages génériques relatifs au serveur collaboratif (agent générateur) : identification, connexion, déconnexion, passage d'un état à l'autre (de cooordination vers production par exemple). Ce client sert également à lancer les clients `ClientComm`, `ClientCoor` et `ClientProd` pour les agents, et ce de manière transparente pour l'utilisateur. Les membres du client `ClientColl` sont représentés en figure 4.10, qui regroupe l'ensemble des éléments des interfaces utilisateurs ainsi qu'un thread d'écoute réseau. Chaque client est étroitement lié à son interface, chacun pouvant faire appel à l'autre. A noter que les clients des agents Communication et Coordination partagent une même interface graphique.

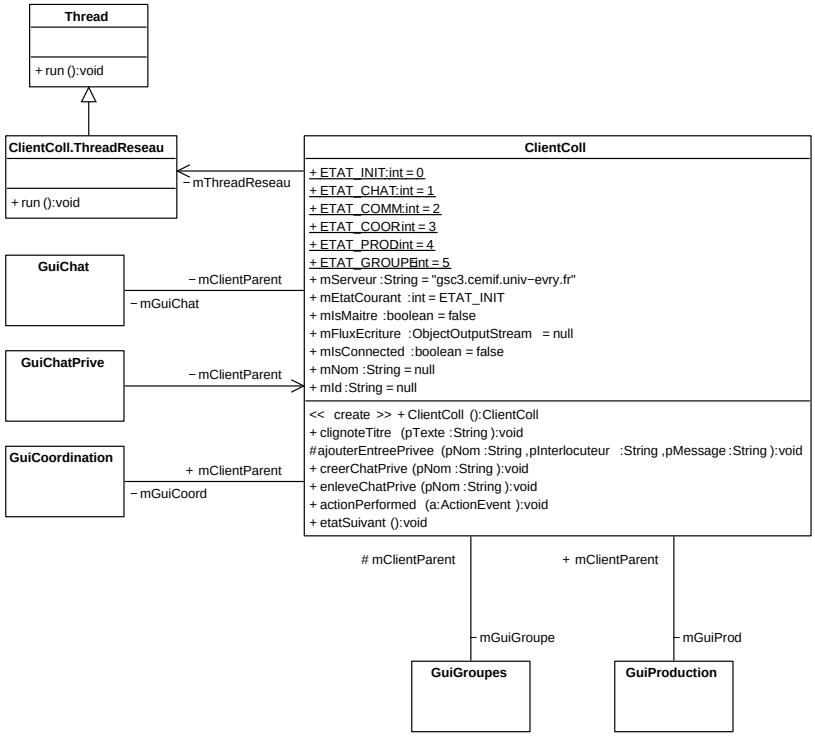

Figure 4.10: Diagramme de classes du client `ClientColl`.

L'architecture résultante est représentée sur le diagramme en figure 4.11.

Elle est constitué de quatre clients réseau et de leurs interfaces et la manipulation du robot ce fait à travers la classe `Action`.

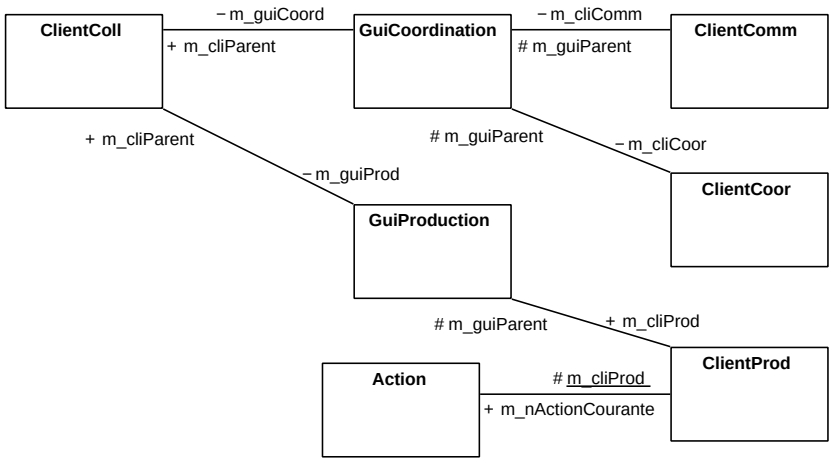

Figure 4.11: Diagramme pour l'architecture du client.

4.3.3 Serveur collaboration

Le serveur est essentiellement constitué de 5 classes d'agents. Ces agents héritent d'une superclasse `agentGenerique` (voir figure 4.12) qui fournit les méthodes de base pour communiquer avec d'autres agents et pour récupérer des informations du *Directory Facilitator* (DF)[5]. Cette classe hérite elle-même de la classe `Agent` offerte par l'environnement JADE.
Tout comme les clients, les agents utilisent les méthodes d'envoi de trames proposées par JADE, en envoyant des messages sur le réseau dans des threads avec exclusion mutuelle sur les sockets de communication.
Nous décrivons brièvement les cinq classes d'agents du SMA-C :

- `Serveur` (agent générique `Gene` dans la figure 4.8) : c'est l'agent le plus complexe, il est serveur réseau pour le chat, client pour la communication avec la base de données, client pour le serveur de commande du robot réel et il gère la création et la coordination des autres agents du système. D'un point de vue global, l'architecture du serveur est représentée en fig 4.13. Outre ses membres et méthodes, le serveur est composé de divers threads destinés au réseau, de plusieurs comportements d'écoute et enfin de tous les agents qu'il a générés.

[5]Annuaire des pages jaunes de JADE

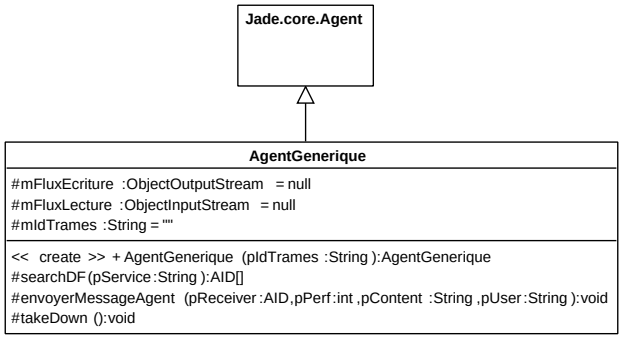

Figure 4.12: Diagramme de classes pour `AgentGenerique`.

- `AgentCollaboration` : il est le noeud central d'un agent collaborateur, gérant la communication entre sous-agents. C'est le seul agent qui ne soit pas un serveur réseau, n'ayant rien à échanger avec le client distant. Cet agent, qui est le noeud de communication d'un agent collaborateur, a pour principale fonction de stocker les adresses des différents agents avec lesquels il communique : agent générateur (serveur), agents communication, coordination et production. L'agent collaboration, comme les autres agents du groupe, utillise le DF pour retrouver les identifiants des agents du groupe, connaissant le service sous lequel ils sont enregistrés. Voir figure 4.14, sa principale fonction est d'être un noeud de communication.

- `AgentCommunication` : il participe aux échanges entre les clients du groupe pendant la phase de communication, notamment pendant la phase de choix de mission. La figure 4.15 présente son diagramme de classes. Il possède un thread d'écoute réseau et trois comportements d'écoute, pour trois classes d'agents différentes.

- `AgentCoordination` : c'est l'agent qui coordonne le choix des actions et synchronise les clients pendant la production. Cet agent se comporte différemment selon qu'il est client maître ou non. S'il est maître, il recense les choix d'actions effectués par les différents clients et répartit les actions entre les clients une fois que le maître a validé les choix. Le maître a également pour rôle de synchroniser les agents production (dès qu'une fin d'action est notifiée, il envoie un message de passage à l'action suivante). Dans le cas d'un client non maître, l'agent relaie toutes les informations reçues du client à l'agent

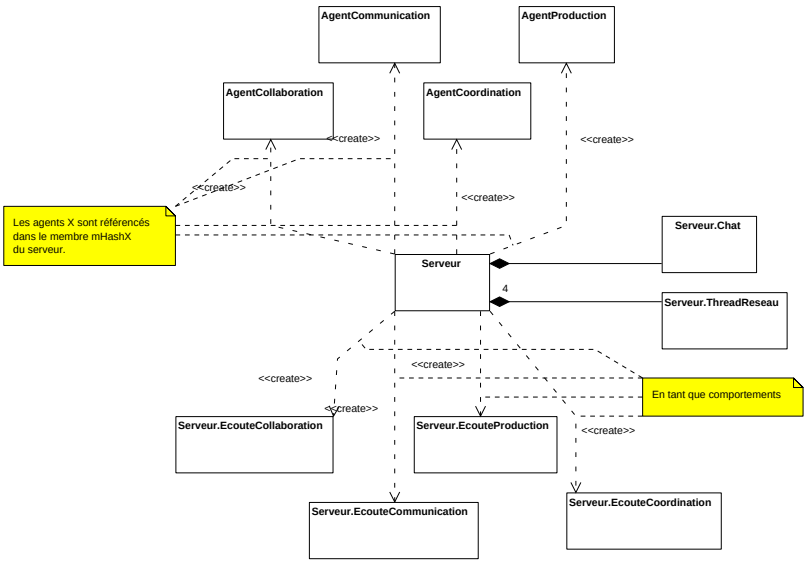

Figure 4.13: Vue globale de la classe Serveur

maître. La figure 4.16 présente son diagramme de classes. Comme pour l'agent communication, il possède un thread d'écoute réseau et trois comportements d'écoute.

- `AgentProduction` : Cet agent notifie le client de l'étape de production courante, la position du robot (supervision) et lit l'étape de production courante du client (en cas de client en action). Son diagramme de classes est présenté en figure 4.17.

4.3.4 Communication entre entités

Cette section présente le protocole utilisé pour communiquer entre clients et agents ainsi que les agents entre eux. Dans un premier temps, le protocole d'identification des clients `ClientColl` est présenté, suivi par les différents protocoles utilisés à mesure que les clients passent d'un état à l'autre.
Il est à noter que le protocole du SMA-C présenté dans ce chapitre a été adapté pour convenir aux exigences de cette application. Il n'est plus question ici d'automatiser le choix de missions ou d'actions, ce sont les utilisateurs qui décident. Il n'y a donc plus d'utilisation de dialogues en *Contract Net* ou *Rational Effect*. De même, comme un seul groupe peut manipuler le robot à la fois, il a été choisi par simplicité de ne permettre que la création d'un groupe à la fois, simplifiant également l'administration du système.

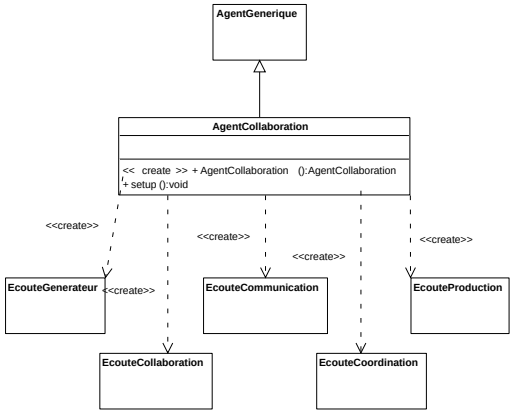

Figure 4.14: Diagramme de classes d'un agent Collaboration.

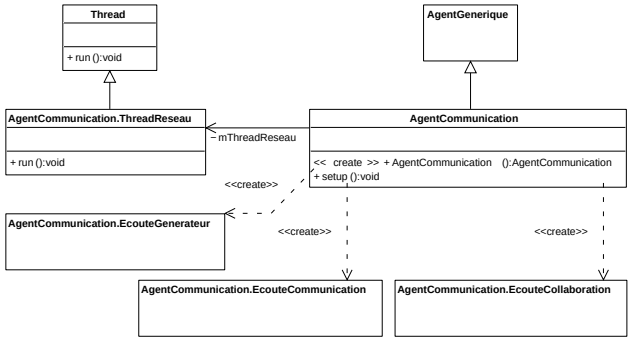

Figure 4.15: Diagramme de classes d'un agent Communication.

Le SMA ici présenté est donc une version simplifiée du SMA testé et validé dans le chapitre précédent.

4.3.4.1 Identification des clients, composition du groupe

La première chose que doit faire un utilisateur à la connexion est de s'identifier, à savoir envoyer son login, son mot de passe et le nom qu'il souhaite avoir, par la classe `ClientColl`. Après vérification dans la base de données que l'utilisateur est bien identifié, après vérification qu'il n'y a pas déjà quelqu'un présent sous le même login et/ou le même nom, un message est retourné pour signaler au client qu'il peut continuer la communication. Dans le cas contraire, un message d'erreur est retourné, suivi de la fermeture des ressources ouvertes pour l'identification, au niveau serveur.
Dès qu'un client arrive, change son nom ou se déconnecte, un message

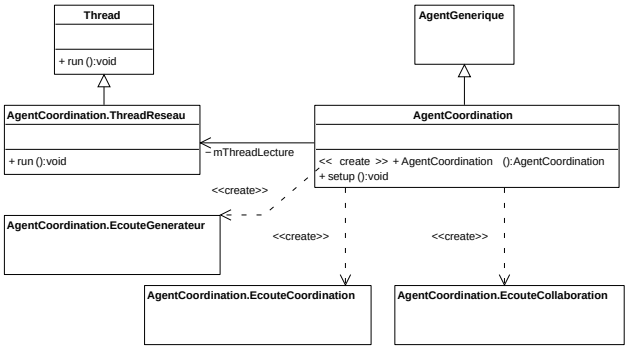

Figure 4.16: Diagramme de classes d'un agent Cooordination.

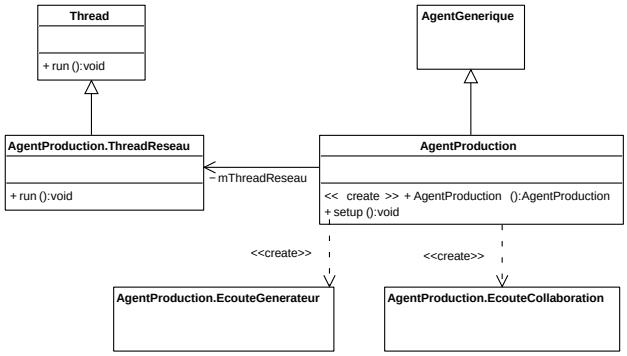

Figure 4.17: Diagramme de classes de l'agent Production.

`clients` est envoyé à tous les clients présents pour qu'ils mettent à jour leurs listes de clients connectés. Ce message permet aussi au client et donc à l'utilisateur de savoir s'il est maître ou esclave.
Un utilisateur maître est le premier utilisateur connecté. Il gère la création d'un groupe, la validation des différentes missions et actions et il a le droit d'exclure un autre utilisateur du groupe. Tout autre utilisateur est esclave. Pour le maître, une interface utilisateur spécifique est disponible pour qu'il compose un groupe. En envoyant la composition du groupe au serveur, le client (logiciel) passe le serveur en attente de connexion des clients `Client-Comm` du groupe pour le passage à l'étape de communication : un message est envoyé à tous les clients `ClientColl` du groupe, qui eux-mêmes connectent les clients `ClientComm`. l'étape de communication est lancée. Le groupe est créé dans la base de données et son maître enregistré.

Les messages de type chat et chat privé sont transmis de manière similaire, le serveur relayant aux intéressés les messages de chat. Un scénario-type est présenté par le diagramme de séquence en figure 4.18. L'échange-type effectué lors de la phase précédent le passage en communication. Un premier client se connecte et s'identifie avec succès. Etant seul, il devient le maître de la session. Un deuxième client arrive et fait de même. Après chat (non représenté pour des raisons de lisibilité), le client maître envoie un groupe constitué des clients 1 et 2 au serveur, qui enregistre l'information et invite les deux clients à passer à l'étape suivante. Dans la suite, on regroupera les différents threads du serveur dans la même entité "Serveur", par simplicité.

4.3.4.2 Etape de communication

Pour activer la communication, le serveur doit attendre que tous les clients `ClientComm` attendus dans le groupe se connectent. Quand un client se connecte, si l'identification s'est bien effectuée, un agent collaboration et un agent communication sont créés. Chacun reçoit en paramètre le nom et l'id du client ainsi que l'adresse de l'agent avec lequel il doit communiquer (collaboration pour communication et vice-versa). Une fois le groupe complet, l'agent collaboration du maître reçoit les adresses des autres agents collaboration, et de même pour son agent communication qui reçoit les adresses des autres agents communication. Enfin, la liste des missions est envoyée à l'agent collaboration maître qui relaie à tous ces informations, jusqu'aux clients communications, qui s'activent alors. Ce processus est décrit dans le diagramme en figure 4.19. Ce diagramme de séquence présentant les divers messages transitant lors du lancement de l'étape de communication, avec deux clients attendus dans le groupe. Le client 1, associé à l'agent collaboration qui reçoit les missions du serveur, est le maître. A la fin de ce processus, les agents peuvent communiquer entre eux, chacun ayant la liste des missions ainsi que la liste de ses interlocuteurs dans le groupe.
Une fois la communication lancée, le chat comme le choix de mission transitent par les agents communication du groupe, chacun relayant ce qu'il reçoit de son client vers les autres agents. Cependant, le choix de la mission ne peut être fait que par l'utilisateur maître. La mission choisie est également retournée au serveur, pour enregistrement dans la base de données. Ce processus est présenté dans le digramme en figure 4.20. La communication entre agents communication est simple, chaque agent relayant les messages qu'il reçoit de son client à la liste de ses interlocuteurs. Cependant, seul le client maître peut relayer un choix de mission.

4.3.4.3 Etape de coordination

L'initialisation de l'étape est similaire à celle de la communication. Quand un client `ClientCoor` attendu se connecte et s'identifie, un agent coordination est créé, son agent collaboration récupère son adresse. Une fois créé, l'agent coordination maître est notifié par le serveur de son statut particulier. Une fois que tous les agents attendus dans le groupe se sont identifiés auprès du serveur, la liste des missions est envoyée à chacun d'entre eux. Enfin, le maître reçoit la liste de ses interlocuteurs (agents coordination du groupe). Ce signal lui fait envoyer à son client la liste des actions et à tous les autres agents coordination sa propre adresse, qu'ils sachent qui notifier des choix de leurs clients. Une fois que ces agents ont reçu l'adresse du maître, la coordination peut commencer et ils envoient la liste des missions à leurs agents. Le diagramme correspondant est représenté en figure 4.21. Comme précédemment, dans ce diagramme le groupe est constitué de deux clients dont le premier est maître. Après connexion et création de tous les agents, chaque agent reçoit sa liste de missions, l'agent maître reçoit la liste de ses interlocuteurs et la coordination commence, activée au niveau client par la réception de la liste des actions.
Pendant l'étape de répartition des actions entre les clients, deux cas de figure se présentent pour l'agent coordination : s'il n'est pas maître, il relaie les actions choisies par son client au maître. Le maître détermine alors si l'action peut être associée à ce client et l'associe alors. Tous les agents sont alors notifiés du statut courant des choix (qui a choisi quoi). Si c'est le client maître qui choisit une action, celle-ci est traitée directement par son agent, et comme précédemment le statut est renvoyé à tous.
Quand le client maître valide les actions, si toutes sont attribuées, alors le passage en production est lancé. Dans un premier temps, chaque agent coordination reçoit les actions que son agent production associé devra effectuer. Cette information est remontée à l'agent collaboration correspondant. L'agent collaboration aura alors pour tâche, à la réception d'un message disant de passer à une action x, de déterminer si cette action devra être réalisée ou supervisée par son agent production. Enfin, une fois ces messages transmis, le message de passage en production est relayé jusqu'aux clients, qui devront alors lancer leurs clients productions pour passer à la dernière étape.
Un exemple de ce type de communication est présenté sur le diagramme de séquence en figure 4.22. Les deux clients précédemment évoqués s'échangent leurs choix de mission via l'agent coordination maître. Une fois que le

maître a décidé de valider les actions, ces dernières sont réparties à chacun, en prévision de la bonne marche de l'étape de production.

4.3.4.4 Etape de production

La création des agents production suit le même fonctionnement que pour les autres agents : les clients `ClientProd` se connectent, s'identifient et une fois tous les agents créés, une fois que ceux-ci connaissent les adresses de leurs interlocuteurs, un signal est envoyé. Ce signal est envoyé à l'agent coordination maître, qui va ensuite envoyer le code de la première action à son agent collaboration. Ce dernier relaie cette information à tous les agents collaboration du groupe. Quand un agent collaboration est notifié de l'action qui doit commencer, il vérifie s'il a cette action dans sa liste d'actions à effectuer. Si c'est le cas, il envoie un message de passage à l'action à son agent production. Sinon ce dernier reçoit un message de supervision d'action.
Une fois l'action lancée, le client production qui fait l'action courante envoie régulièrement à son agent production le statut de la production, notamment la position du robot et l'état des cylindres manipulés. Cette information est relayée à tous les agents production du groupe via les agents collaboration. Ainsi les clients production reçoivent les coordonnées du robot et peuvent suivre l'avancée de la production. Une fois que l'agent production reçoit un message de fin de production de son client, l'information est transmise à tous les agents coordination via les agents collaboration. L'agent coordination maître lance alors l'action suivante ou, s'il n'en reste plus, envoie le signal de fin de mission au serveur. Ce dernier réinitialise alors l'état des clients, tue les agents, vide le groupe, un nouveau cycle peut reprendre. Quand l'agent coordination 1 a reçu du serveur le message de lancement de production, il signale à tous les agents collaboration l'action en cours, jusqu'aux clients. Le client de production envoie un statut du robot pour la supervision et des commandes pour le robot réel. Une fois la production terminée, la mission est finie et les agents détruits pour recommencer un nouveau cycle de collaboration.

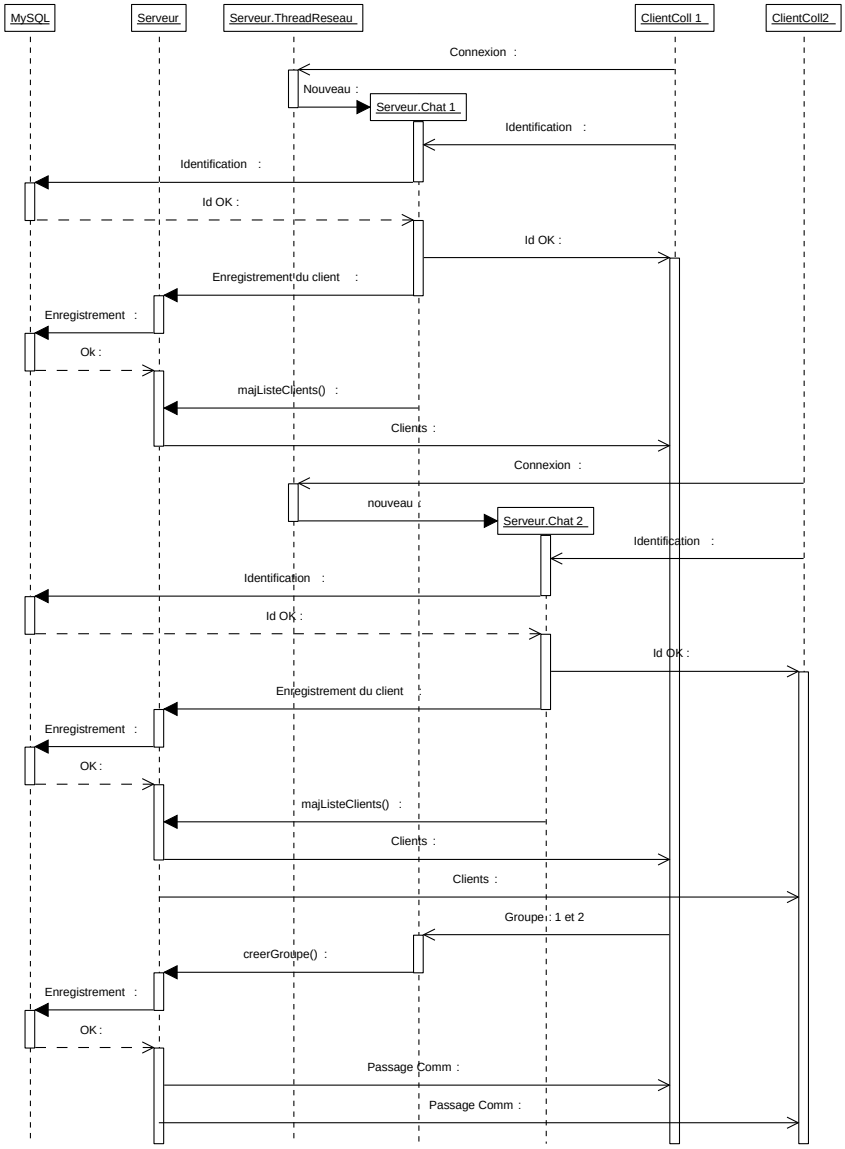

Figure 4.18: Un scénario-type.

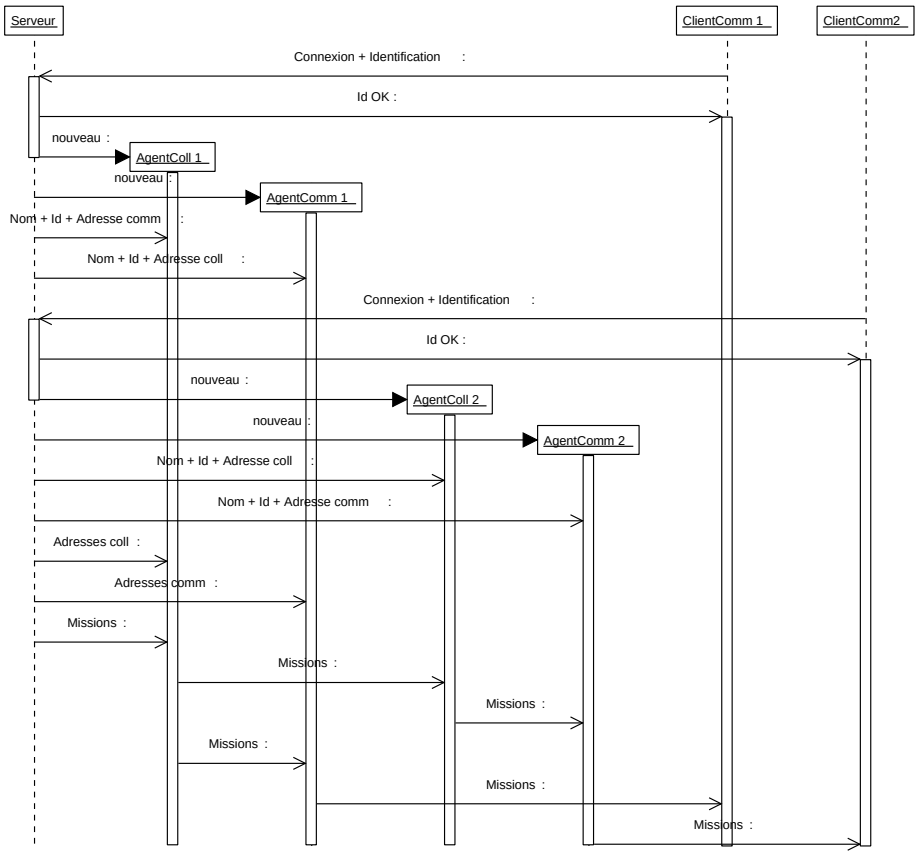

Figure 4.19: Diagramme de séquence de la communication.

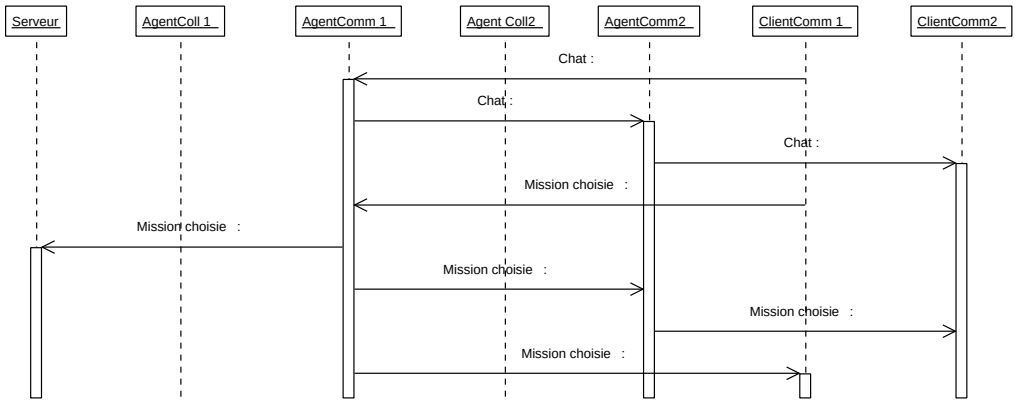

Figure 4.20: Diagramme de séquence du choix de la mission

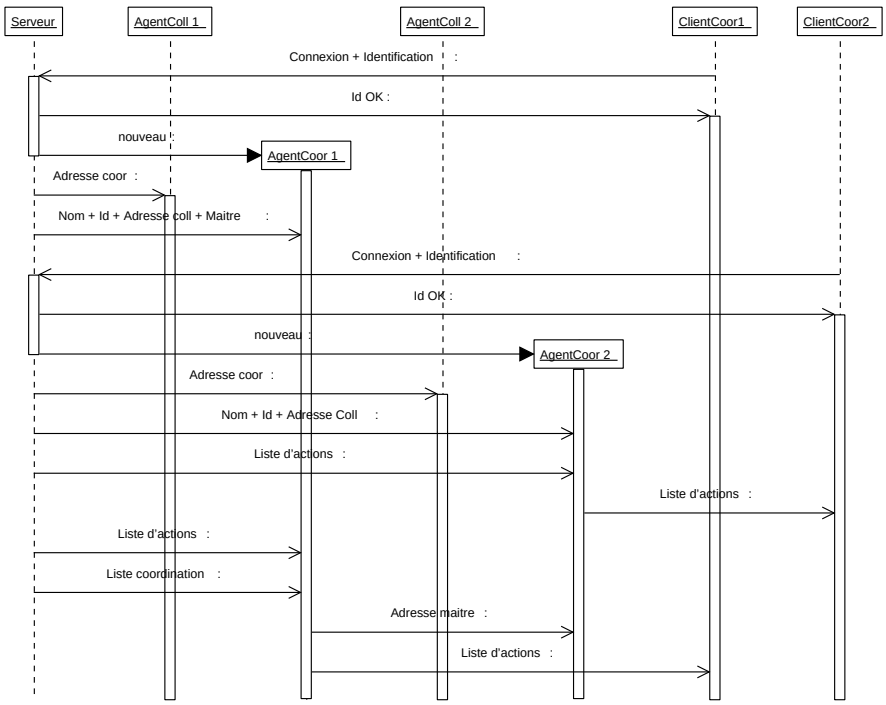

Figure 4.21: Diagramme de séquence du choix des actions.

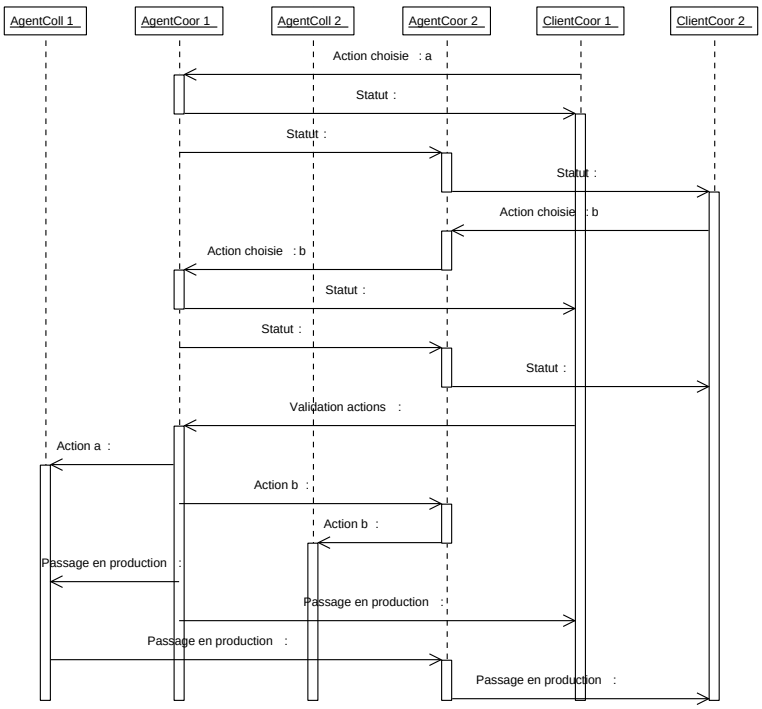

Figure 4.22: Diagramme de séquence de la validation des actions.

4.4 Interface Homme-Machine de ARITI-C

Cette section présente les différentes interfaces offertes à l'utilisateur pour chacune des fonctionnalités d'ARTI-C.

4.4.1 Interface principale

Cette interface est l'héritage de l'ancien système Ariti : c'est celle qui s'affiche au lancement de l'applet, elle est le noeud autour duquel sont conçues les autres interfaces pour les fonctionnalités annexes (figure 4.23). L'interface est décomposée en 3 vues et un panel de contrôle. Le panel de contrôle est constitué d'onglets correspondants à diverses fonctionnalités et d'un bouton "Connexion" destiné à ouvrir l'interface de travail collaboratif. La vue en haut à gauche est le retour vidéo du robot réel sur laquelle on superpose le robot virtuel. La vue en haut à droite est paramétrable, par défaut elle affiche une vue de côté de l'environnement virtuel. Enfin la vue en bas à gauche est une vue de l'environnement du point de vue de l'effecteur (la tige).
Le premier onglet, activé en figure 4.23, est celui du contrôle de la vue du robot virtuel. Les différents boutons ont les fonctions suivantes :

- Socle : affiche le modèle du socle du robot.
- Support : affiche le modèle du support du robot.
- Grande base : affiche le modèle de la grande base du robot.
- Equerre : affiche le modèle de l'équerre du robot.
- Tige : affiche le modèle de la tige du robot.
- Voir robot : affiche ou désactive l'affichage de l'intégralité du robot.

Cette interface possède également 3 autres onglets. Le premier onglet offre le contrôle de la caméra virtuelle en haut à droite de l'interface et permet également de lancer l'interface de télécalibration de la caméra (voir annexe B). Le second onglet permet de contrôler le robot en production et le dernier permet de lancer l'interface de création de guides (voir annexe C).

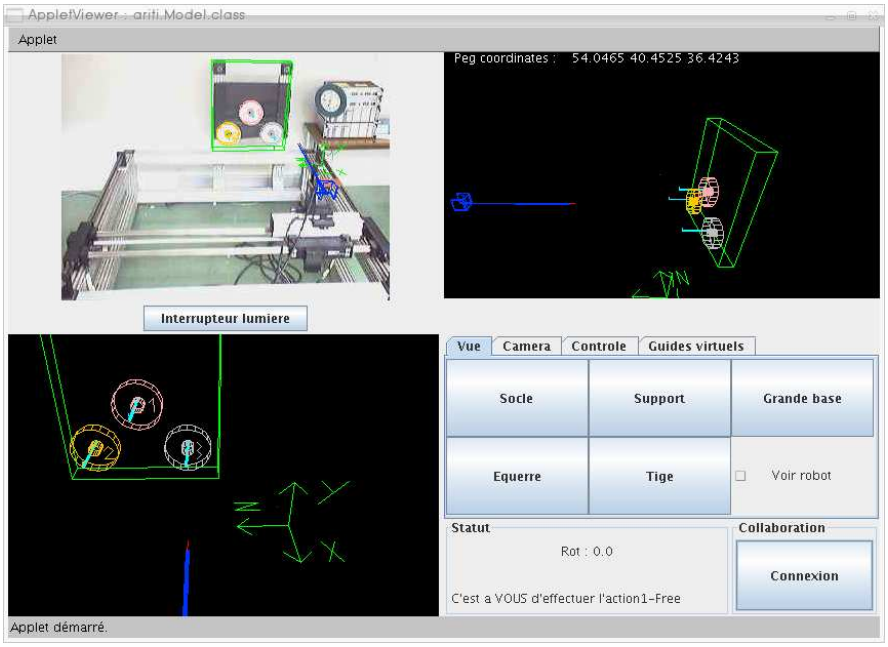

Figure 4.23: Interface principale avec ses 3 vues et son panel de contrôle

4.4.2 Interface de collaboration

Dans cette partie, nous présentons l'interface homme-machine utilisée pour la téléopération collaborative [KHE 05d] à travers une manipulation-type faisant intervenir deux utilisateurs pour une mission de saisie-dépôt.

4.4.2.1 Enregistrement

Tout utilisateur d'Ariti doit être connu du système. Pour cela une fenêtre s'ouvre dès la connexion d'un utilisateur, lui demandant de préciser son identifiant et son mot de passe ainsi que le pseudonyme sous lequel il veut être connu des autres usagers (figure 4.24). Si l'utilisateur n'est pas enregistré, un clic sur le bouton approprié lui permet d'accéder à l'interface d'enregistrement, où il précise divers renseignements (figure 4.25). Son adresse e-mail lui permettra de récupérer ses paramètres de connexion dans la fenêtre appelée par un clic sur "Identifiants perdus ?" (figure 4.26). Ses paramètres de connexion lui seront renvoyés par e-mail.

4.4.2.2 Discussion et création du groupe

Le premier élément d'interface apparaissant après la connexion au serveur collaboratif est l'interface de discussion (*chat*) : voir figure 4.27. Cette

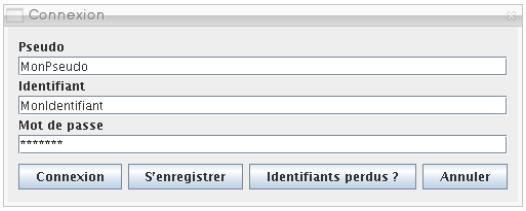

Figure 4.24: Interface de connexion de l'usager

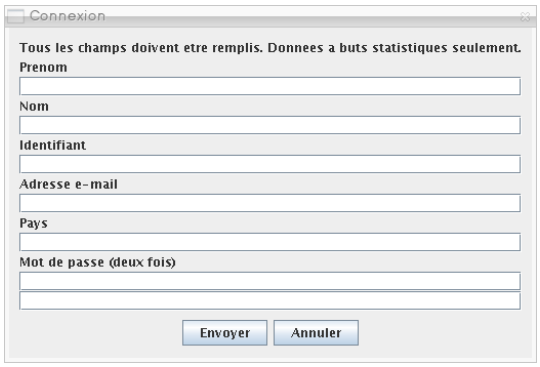

Figure 4.25: Interface d'enregistrement de l'usager

interface reprend certaines caractéristiques des classiques clients IRC. Par défaut, le premier utilisateur connecté est maître de session : il compose le groupe et valide ses choix. Il peut également exclure des utilisateurs. Son interface possède donc quelques boutons de plus que les autres utilisateurs. La liste des utilisateurs est donnée à droite de l'interface, permettant des interactions telles qu'un dialogue en privé ou une sélection d'un utilisateur pour l'exclure ou lui transférer les droits de maître. Le maître est identifié par un astérisque à la fin de son nom. Les membres du groupe courant sont identifiés par une arobase. Les boutons de l'interface sont les suivants (disponibles uniquement pour le maître) :

- Reinit : réinitialise la session : tous les utilisateurs reviennent en mode de discussion, la mission est interrompue.
- Composer le groupe : Passe en mode de composition du groupe.
- Rendre select. maitre : transfère les droits de maître à la personne sélectionnée dans la liste.
- Exclure select. : exclut la personne sélectionnée dans la liste.

Figure 4.26: Interface de récupération de paramètres

Le passage en composition de groupe se manifeste chez le maître par l'ouverture d'une nouvelle fenêtre (figure 4.28), contenant deux listes de noms d'utilisateurs : celle des personnes hors du groupe et celle des personnes dans le groupe (vide par défaut). Deux boutons permettent de transférer les personnes d'une liste à l'autre. Le maître sera forcément intégré au groupe. Après validation, la phase de communication commence effectivement au sein du groupe.

4.4.2.3 Communication et coordination

Une fois le groupe constitué (dans les captures d'écran, les deux personnes connectées font partie du groupe), chaque membre a une nouvelle interface qui apparaît, constituée d'une zone de discussion intra-groupe (à gauche) et d'une liste déroulante de missions (à droite) : figure 4.29. Le maître a des boutons de validation ou de choix automatique (aléatoire) de mission. Une fois la mission choisie, la zone de choix de mission est remplacée par une zone de répartition d'actions (figure 4.30) : à chaque action est associé un bouton. Chaque utilisateur cliquant sur un bouton réserve l'action correspondante si elle n'est pas réservée. Un deuxième clic libère l'action. Chacun est averti des choix des autres utilisateurs par un texte dans la zone centrale de l'interface. Le maître ne peut valider que quand toutes les actions sont distribuées. Il peut également choisir d'un clic sur un bouton une répartition automatique des actions. La validation amène au passage en production.

4.4.2.4 Production

Le passage en production est accompagné de l'ouverture d'une dernière fenêtre dans l'interface. Celle-ci affiche le statut de la production (qui fait quoi) : figure 4.31. Une liste déroulante permet d'activer ou non les guides (voire de ne pas tenir compte de ces derniers), un bouton permet d'automatiser la tâche (en missions de saisie / dépôt seulement) et un

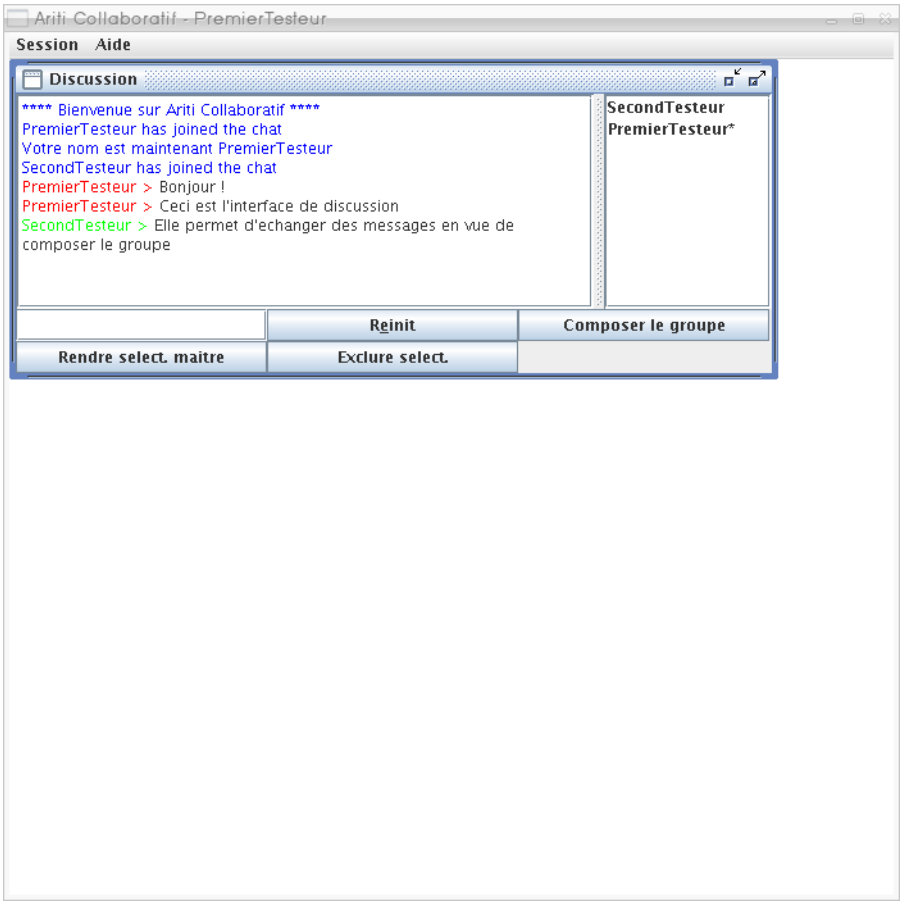

Figure 4.27: Interface de discussion

dernier bouton permet de signaler que l'on a fini son action au cas où le système ne le détecterait pas (il est envisageable également qu'un utilisateur souhaite ne pas se plier au contraintes de son action).
Dans le cadre des missions de saisie / dépôt, la production se déroule au niveau de l'interface principale. En utilisant le panel de contrôle du robot ou les raccourcis clavier, l'utilisateur qui fait l'action dirige le robot virtuel, avec l'assistance des guides. L'utilisateur en supervision voit le statut de la production sans avoir les aides visuelles de l'autre utilisateur. Enfin, une fois toutes les actions terminées, un message invite le maître à réinitialiser la session (figure 4.32).

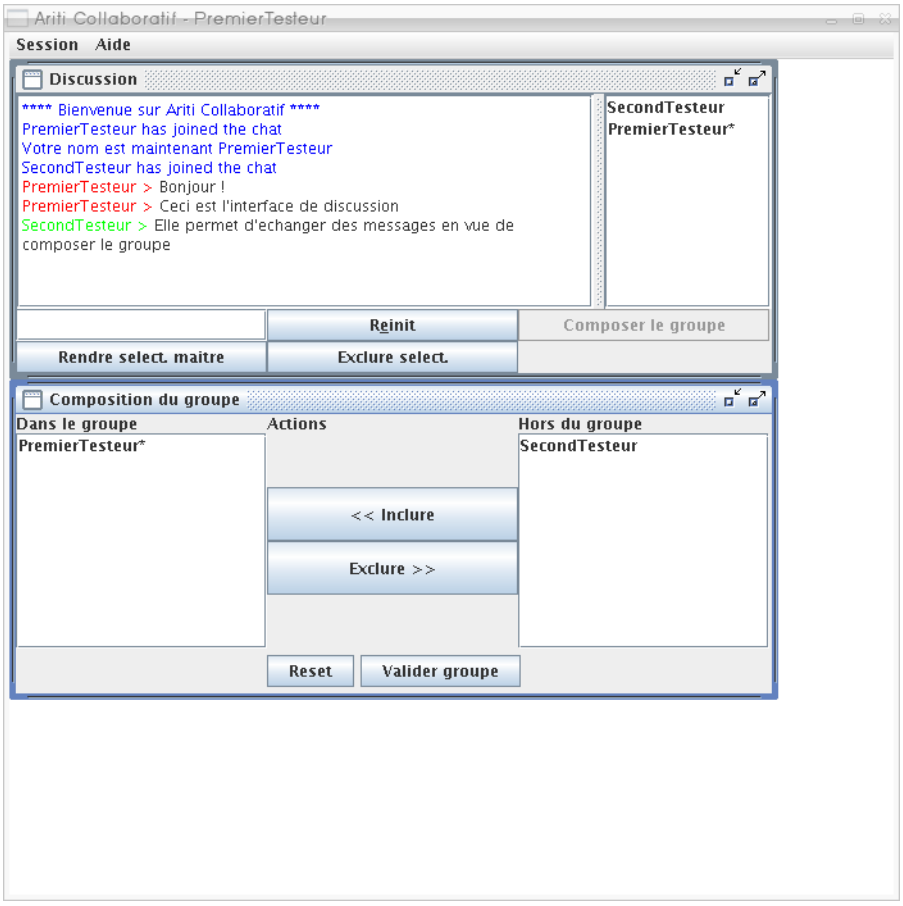

Figure 4.28: Interface de création de groupe

4.5 Description des diverses missions rencontrées

Le système collaboratif d'Ariti sait gérer une poignée d'actions servant de briques de base à la composition des missions. Nous présentons ci-dessous les diverses actions gérées, regroupées par thématiques.

4.5.1 Les missions de saisie / dépôt

Ces missions sont les différentes étapes permettant de déplacer un objet (cylindrique) d'un crochet (support) à un autre. Elles peuvent être réelles (le robot réel est asservi au robot virtuel) ou simplement virtuelles (manipulation du robot virtuel uniquement).

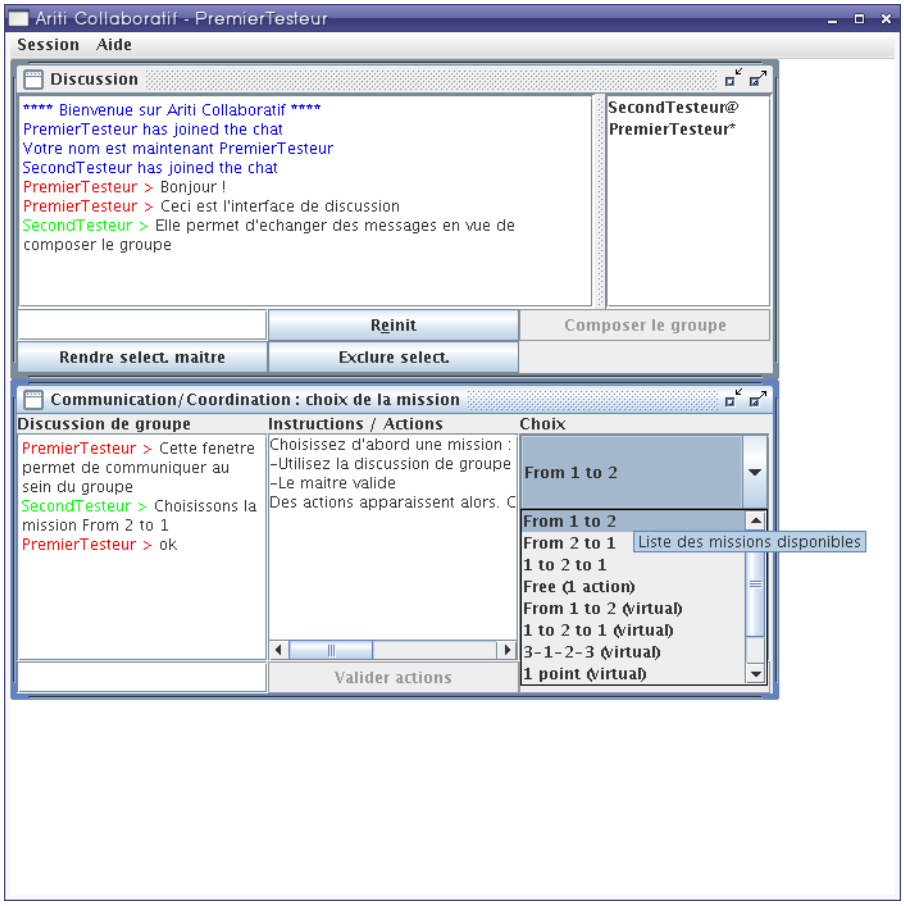

Figure 4.29: Interface de choix de mission

Atteinte de la cible x (Reach on) : Ces actions consistent à amener l'effecteur au contact du premier objet sur le crochet x (1, 2, 3). Le guide associé est un guide cônique dirigeant vers le point visé (figure 4.33-A).

Saisie de la cible x (Grab on) : Ces missions consistent à retirer du crochet x le cylindre préalablement atteint. Le guide est composé d'une série de 3 tubes permettant le retrait du cylindre sans accrocher le support (figure 4.33-B). L'action consiste à d'abord avancer pour piquer le guide puis ensuite le retirer.

Dépôt de cylindre sur x (Deposit on) : Ces missions consistent à déposer un cylindre saisi sur le crochet x. Pour cela, un guide constitué d'un cône et de 3 cylindres permet l'approche puis le dépôt de l'objet (figure 4.33-C).

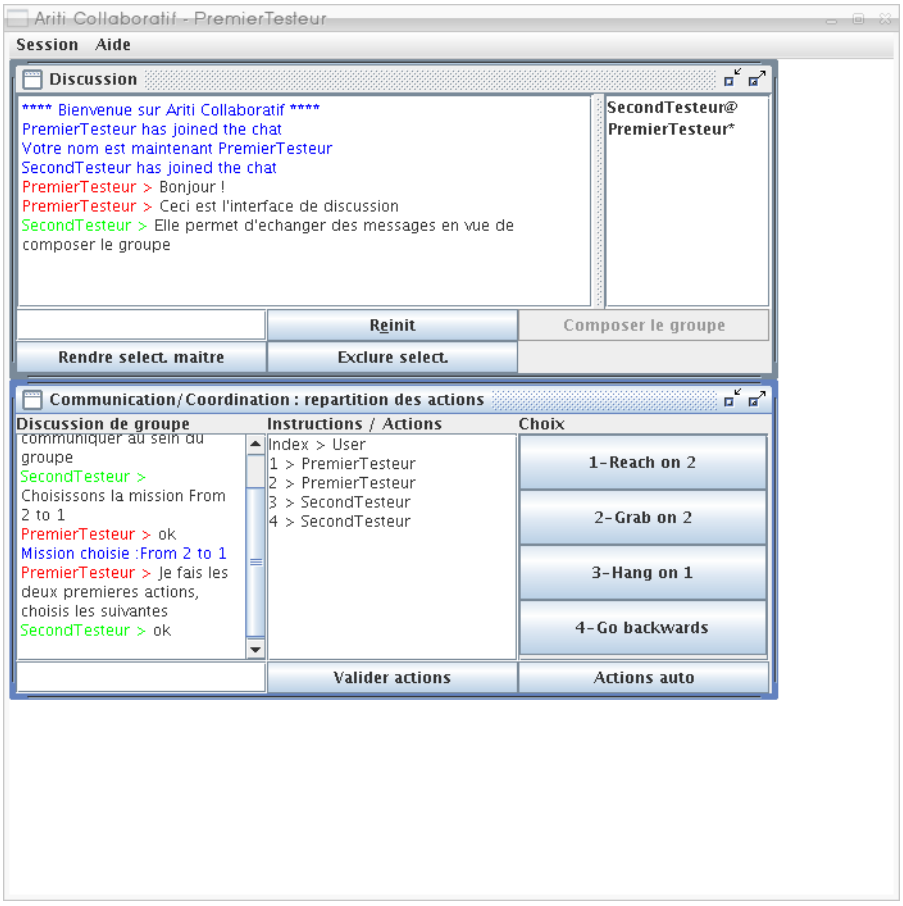

Figure 4.30: Interface avec la liste d'actions

Recul (Go backwards) : Ces missions consistent à éloigner l'effecteur des crochets, pour se préparer à une nouvelle manipulation (figure 4.33-D).

4.5.2 Création collaborative de guides virtuels

Chaque utilisateur crée et manipule des guides dans son environnement de création. Les guides créés et modifiés par les utilisateurs sont visibles par tous. Cependant, les guides utilisés par d'autres ne sont pas sélectionnables, et affichés en blanc pour signaler ceci. La fin de l'action est déterminée quand le maître décide d'insérer les guides dans l'environnement en cliquant sur le bouton **valider**. Les espaces de création de guides sont alors vidés et les guides insérés dans l'environnement d'ARITI-C.

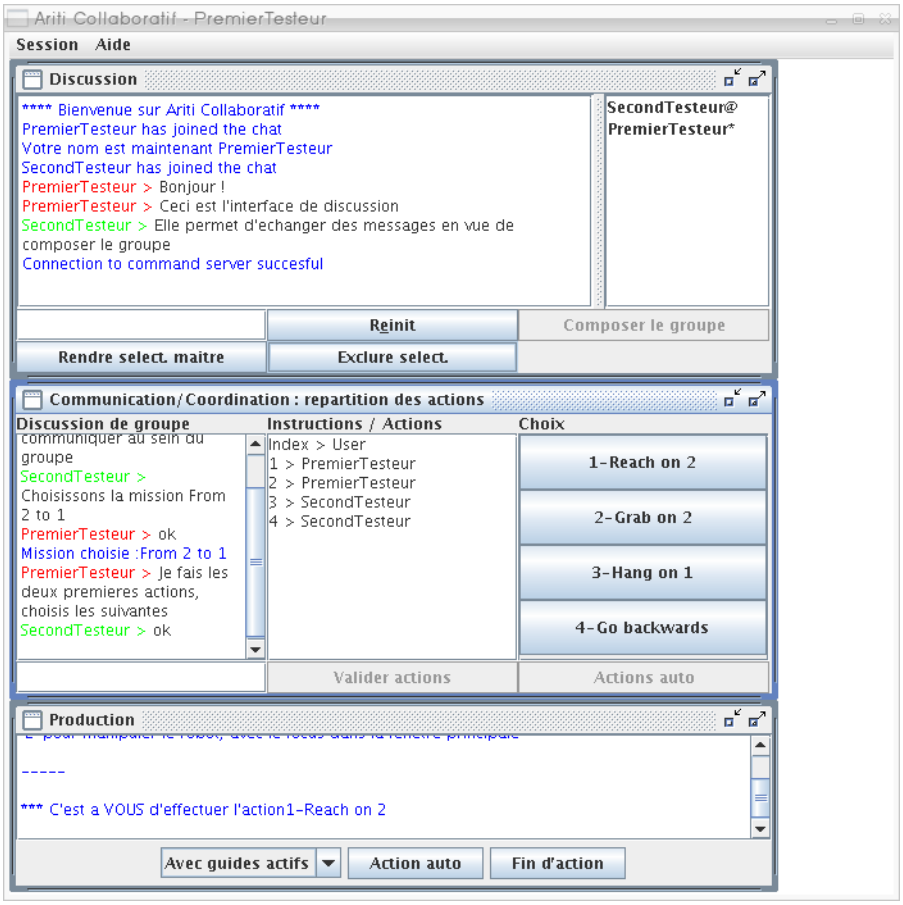

Figure 4.31: Interface lors du passage en production

Nous résumons ci-dessous le fonctionnement du partage collaboratif de guides.

Principes :
La problématique soulevée par le partage des guides est le partage des ressources. Celui-ci doit être assuré au niveau du serveur, empêchant deux utilisateurs d'accéder à une même ressource simultanément. Pour qu'un utilisateur accède à une de ces ressources (un élément de guide), il faut auparavant qu'il le sélectionne. C'est donc ce point que le serveur va contrôler, recensant et autorisant ou non la sélection d'un guide. La logique de conception du SMA-C veut que ce soit un agent coordination qui remplisse cette tâche, qui sera donc associée à l'agent coordination maître du groupe. Comme JADE gère ses agents dans un seul thread, on peut se contenter de tenir à jour une liste d'associations (client, guide) pour assurer qu'un guide

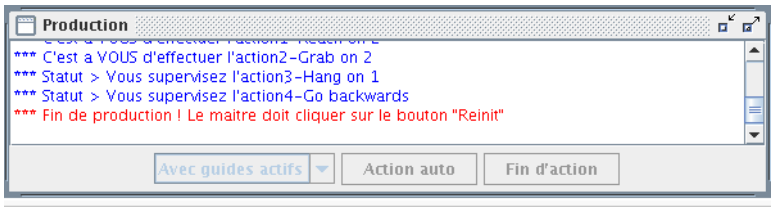

Figure 4.32: Fin de la production

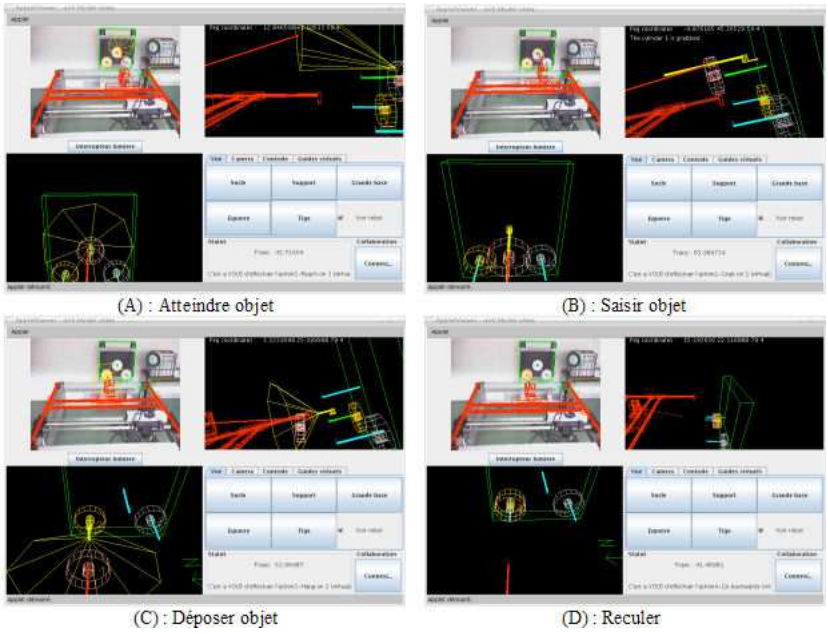

Figure 4.33: Caputure d'écrans d'une mission de saisie / dépot d'un objet en virtuel.

est utilisé par au plus une personne.

Les guides devront donc avoir un identifiant unique. Par simplicité, cet identifiant est constitué de la concaténation de l'identifiant du client ayant généré le guide (son login) et d'un compteur de guides créés par le client, le tout séparé par un tiret. Cet identifiant est donné par le client au moment de la création du guide. Comme le login est unique, aucun guide créé par un autre client n'aura le même identifiant.

Au niveau de l'agent coordination, une table de hachage permet d'associer un identifiant de guide sélectionné à chaque client (identifiant vide si pas de sélection pour le client). La collaboration se résume alors au processus

suivant :

- Lorsque le client crée, supprime, modifie, sélectionne un guide x, il envoie la trame correspondante.
- La trame est acheminée jusqu'à l'agent coordination maître.
- Le traitement idoine est fait par l'agent, mettant à jour ou utilisant la liste des associations (client, guide).
- Une trame est renvoyée à tous les clients susceptibles d'être concernés, qui répercutent sur leur affichage.

Transmission des trames :

Le client utilisé pour la transmission réseau est, logiquement, le client production, qui communique avec un agent production. Il faut donc remonter l'information à l'agent coordination maître. La trame transite donc par l'agent collaboration de l'agent collaborateur et est redirigée vers l'agent coordination. Si celui-ci n'est pas maître, il envoie directement la trame à l'agent coordination maître, dont il connaît l'adresse.

Le retour d'informations depuis l'agent coordination maître est exactement le même en sens inverse.
L'exemple du diagramme de séquence en figure 4.34 illustre cette communication. Il suppose que la collaboration se fait entre deux clients 1 et 2, 1 étant le maître. On suppose qu'avant le diagramme, le client 1 avait sélectionné un guide A et le client 2 un guide B. Le client 2 envoie une requête de sélection d'un guide C. L'agent coordination 1 l'autorise, en lui envoyant une trame de déselection de B et de sélection de C. Le client 1 reçoit une trame informative lui indiquant la disponibilité du guide B et la sélection du guide C par un autre client.

4.5.3 Les autres missions

Ces missions sont conçues pour être utilisées aussi bien indépendamment que au sein d'une mission de saisie / dépôt.

Libre (Free) :
Cette mission donne toute latitude à l'utilisateur pour manipuler le robot,

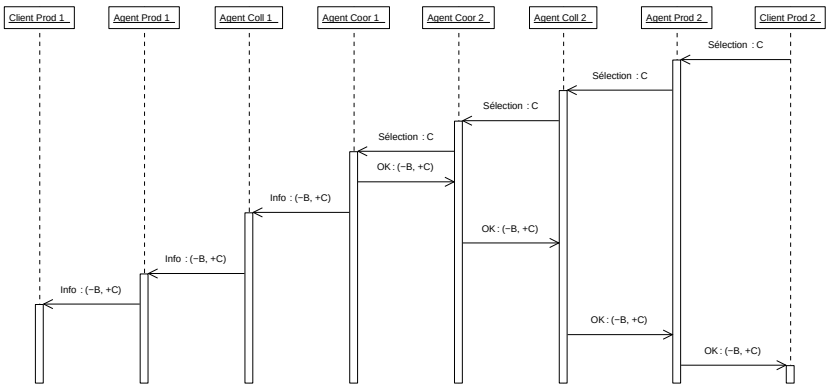

Figure 4.34: Communication-type durant une phase de création collaborative de guides virtuels.

aussi bien virtuel que réel. Il n'y a pas de guides. Pour terminer cette action, il faut cliquer sur le bouton de fin d'action de l'interface de production.

Atteindre point 3D (Reach 3D point) :
Cette mission consiste à atteindre un point que l'utilisateur définit lui-même. En cliquant dans les vues virtuelles, l'utilisateur désigne une droite dans chacune. Le point le plus proche de ces droites est visé. Un guide cônique permet de l'atteindre pourvu qu'il soit dans le domaine accessible au robot (figure 4.35).

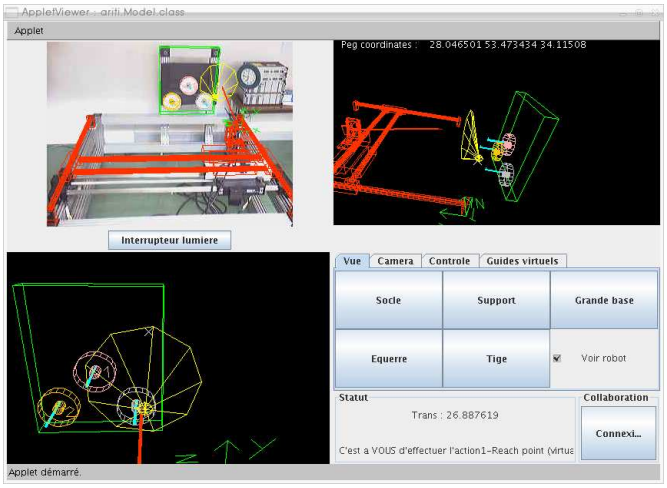

Figure 4.35: Guide apparu après sélection de point 3D par clics successifs dans les deux vues virtuelles.

4.6 Bilan

Un système de téléopération permet à un opérateur de réaliser une tâche à distance, en l'éloignant de l'environnement de travail et des machines qu'il contrôle. Pour aider l'opérateur à réaliser cette tâche plus efficacement, il lui est, parfois, indispensable d'être assisté par d'autres utilisateurs : c'est ce que nous appelons la téléopération collaborative. Dans la première section de ce chapitre, nous avons vu que certains systèmes existants de téléopération, ne permettent pas la communication et la coordination des utilisateurs pour manipuler le robot.
Le but de ce chapitre était d'utiliser le SMA-C développé dans le chapitre précédent pour répondre à ces manques en proposant une solution qui permet à un groupe d'utilisateurs de collaborer pour la manipulation du robot. Nous avons transformé le système mono-utilisateur ARITI en un système multi-utilisateur collaboratif ARITI-C.
Nous avons présenté l'architecture logicielle d'ARITI-C ainsi que la façon dont les utilisateurs se connectent, communiquent, coordonnent et produisent ensemble en manipulant un seul robot en même temps. Nous avons également montré l'intérêt du SMA-C pour la création collaborative de guides virtuels qui sont utilisés comme de véritables outils d'assistance à la téléopération.
Dans le chapitre suivant nous présentons l'évaluation de l'ergonomie de l'interface homme-machine de ARITI-C. Nous présentons aussi l'outil d'analyse de la collaboration développé ainsi que les statistiques d'utilisation.

Chapitre 5

Évaluation de ARITI-C

5.1 Introduction

Le développement rapide des technologies de l'information a provoqué un impact sans précèdent dans le domaine de la collaboration. Les humains ont cherché à effectuer des tâches collaboratives sous des formes qui n'ont jamais été envisagées auparavant. Les Emails, les chat-rooms, vidéoconférence, ... et autres innovations ont ouvert des portes à de nouvelles formes de collaboration et qui a permit d'améliorer efficacement le rendement et la productivité. Cependant, plusieurs perfectionnements sont à envisager spécialement dans l'adaptabilité des outils existant dans les collecticiels, les workflow et dans les modèles de collaboration, par conséquent il est nécessaire d'évaluer ces systèmes collaboratifs.
Une myriade de systèmes et d'outils ont été développés afin de concevoir de différents paradigmes de collaboration. Cela met en évidence le fait que plusieurs questions relatives à la conception et au système d'information mérite des recherches approfondies. L'efficacité de ces démarches dans le développement de nouveaux systèmes peut être constaté par une méthode d'évaluation adéquate. Étant donné la complexité inhérente aux applications collaboratives, leur évaluation reste une tâche très laborieuse. Cependant nous trouvons dans la littérature des travaux intéressants qui traitent le problème de l'évaluation des systèmes. [HAL 00] a présenté une étude sur les types de métrique qui peuvent être collectées pour évaluer les applications collaboratives. [GUT 00] propose une structure qui aide à évaluer le problème de l'utilisabilité en considérant le mécanisme de la collaboration. Notre travail est inspiré des deux travaux cités ci-dessus ainsi que sur nos recherches sur la qualité des logiciels.

5.2 Evaluation des systèmes collaboratifs

L'évaluation des systèmes collaboratifs consiste à estimer les performances du système par rapport à un ensemble d'exigences et aux besoins des utilisateurs. Il existe plusieurs méthodes et techniques pour évaluer ces systèmes. A partir de l'état de l'art établi par [KNU 00] et les travaux de [PIN 00], [DEW 00], nous avons classé ces techniques sommairement de la façon suivante ; les techniques qui utilisent des :

- Évaluations heuristiques
- Tests utilisateur
- Expériences en laboratoire
- Interview et questionnaire
- Simulation
- Scénarios
- Analyse de protocoles et inspection

Il existe d'autres méthodes d'évaluations. La plus élémentaire consiste à effectuer des calculs statistiques. En effet dans toute évaluation, savoir qui utilise le système, à quelle fréquence et dans quel contexte peut s'avérer très pertinent. Un autre type de méthode consiste à calculer des coefficients d'évaluations à partir de la valeur-ajoutée apportée par la collaboration du point de vue économique. Les indicateurs les plus souvent utilisés sont la qualité du produit, le temps d'exécution et le coût. Evidemment, il arrive souvent de combiner ces différentes méthodes pour avoir un résultat plus probant. Etant donné que notre système n'est pas dédié à une tâche industrielle et que le nombre d'utilisateurs n'est pas conséquent pour que l'on puisse effectuer des statistiques, nous avons opté pour l'évaluation de l'utilisabilité de notre système collaboratif.

5.2.1 Utilisabilité des systèmes

L'utilisabilité désigne la qualité d'une application qui est facile et agréable à utiliser et à comprendre. L'utilisabilité d'une application se mesure à trois grands critères objectifs :

- l'efficacité, c'est-à-dire l'atteinte des objectifs par l'utilisateur ;

- l'efficience, c'est-à-dire l'économie des ressources nécessaires pour atteindre ces objectifs ;

- la satisfaction, c'est-à-dire le sentiment d'agrément ou le contentement procuré à l'utilisateur lors de l'utilisation de l'application.

Une application sera donc utilisable si l'utilisateur peut réaliser sa tâche (efficacité), qu'il consomme un minimum de ressources pour le faire (efficience) et que le système est agréable à utiliser (satisfaction). D'autres aspects peuvent entrer en ligne de compte pour évaluer l'utilisabilité générale de l'application, comme la sécurité ou maîtrise (le nombre d'erreurs commises par l'utilisateur et la rapidité de corrections de ces erreurs) et la facilité d'apprentissage (la compréhension correcte et l'assimilation rapide du mode de fonctionnement). Le concept d'utilisabilité a donné naissance à son propre instrument de mesure : les tests d'utilisabilité. Il ne s'agit pas de la seule méthode pour apprendre ce que les utilisateurs pensent de votre application (citons également l'évaluation experte, les interviews, les focus groupes, les données d'usage (statistiques quantitatives ou informations qualitatives)), mais c'est probablement la plus appropriée. En effet, tandis que les interviews ou les focus groupes permettent de recueillir l'opinion des utilisateurs, le test d'utilisabilité permet d'observer directement le comportement de l'utilisateur face à l'application et d'identifier ainsi très concrètement les problèmes qu'il rencontre, car le meilleur moyen de savoir si ce que l'on propose à un utilisateur lui convient, c'est de le lui demander, et le meilleur moyen pour lui de répondre à la question, c'est de l'essayer.

L'objectif du test d'utilisabilité est de demander à un petit nombre d'utilisateurs représentatifs de réaliser quelques tâches type de l'application. C'est en observant l'utilisateur en situation que l'on pourra relever les difficultés qu'il rencontre, les erreurs qu'il commet, les questions qu'il se pose et les fonctionnalités qu'il apprécie ou non. Le principe de base du test d'utilisabilité est de se placer dans un contexte le plus proche possible de l'utilisation réelle. Un observateur donne plusieurs consignes précises à un utilisateur, qui va devoir accomplir quelques tâches particulières. Pour chacune de ces tâches, des mesures sont réalisées pour vérifier si les critères d'utilisabilité sont atteints : l'utilisateur a-t-il pu accomplir la tâche demandée? L'a-t-il fait rapidement et facilement A-t-il commis des erreurs? A-t-il trouvé l'application agréable?
Le rôle de l'observateur est de répertorier et de noter toutes les erreurs

commises, les stratégies de récupération, les incompréhensions (toutes les difficultés d'utilisation rencontrées). Une fois le test terminé, ces différentes remarques servent de base à une " analyse à chaud " avec l'utilisateur afin de mieux cerner les causes des problèmes.

5.2.2 Mise en œuvre de la méthode

Le test d'utilisabilité requiert des ressources matérielles limitées, mais une préparation soignée. Outre la désignation de l'observateur et le recrutement des utilisateurs représentatifs, les ressources suivantes sont indispensables pour conduire un test d'utilisabilité :

- un poste de travail ;
- un scénario ;
- un questionnaire pré-évaluation ;
- une feuille d'observation ;
- une liste de consignes aux observateurs ;
- un questionnaire post-évaluation ;
- une liste des tâches à effectuer.

Lors de la préparation d'un test d'utilisabilité, il faut :

- identifier et recruter les utilisateurs représentatifs ;
- identifier les tâches représentatives
- un scénario, dont le but est de garantir que tous les participants seront traités sur un pied d'égalité. Le scénario doit comprendre une explication des objectifs du test aux participants, une description du déroulement du test et des tâches à effectuer, les consignes à donner aux participants ainsi que le questionnaire qui leur sera remis après le test ;
- un questionnaire pré-évaluation, pour s'assurer que les participants correspondent bien au profil des utilisateurs représentatifs ;
- la liste des tâches à accomplir et les critères requis pour qu'une tâche soit considérée comme accomplie ;

- une liste de consignes aux observateurs, à rappeler avant le début du test ;
- des feuilles d'observation, pour noter les horaires, les actions des participants, les problèmes et les commentaires ;
- un questionnaire post-évaluation, pour mesurer la satisfaction et la compréhension de l'utilisateur et pour recueillir toute autre information que le participant voudrait livrer ;

Il est inutile de multiplier exagérément le nombre de tests. Même un seul test peut apporter de précieux enseignements. On considère généralement que 5 à 6 utilisateurs suffisent à identifier la majorité des problèmes d'utilisabilité. En effet, les problèmes étant liés à l'application et non aux utilisateurs, ce n'est pas en multipliant les testeurs que l'on trouvera plus de problèmes. C'est pourquoi l'on estime qu'augmenter le nombre d'utilisateurs augmente les coûts du test mais pas la pertinence des résultats. Ainsi, plutôt que de mener un test avec quinze utilisateurs, il est préférable de faire trois tests auprès de cinq utilisateurs en améliorant l'application à chaque itération. En outre, la durée d'un test d'utilisabilité ne doit pas excéder environ une heure. Au delà, la fatigue et la lassitude se feront sentir chez le participant et risqueront de perturber l'évaluation de l'application. Il faut tenir compte de ce paramètre lors de l'élaboration de la liste des tâches. On veillera également à vérifier lors des tests préalables que le timing planifié est réaliste.

5.3 Evaluation de l'ergonomie de l'interface homme-machine d'ARITI-C

Lors des tests au laboratoire nous avons soumis un panel de testeurs ne connaissant pas le système à une même manipulation faisant intervenir tous les aspects d'une mission-type. Les testeurs sont des doctorants dans notre laboratoire, leurs spécialités varient entre automaticiens, informaticiens, électroniciens et roboticiens. Pour des raisons techniques (machines de même caractéristiques) tous les testeurs ont manipulé le robot dans une même salle (la salle evr@). Ils ont tous respectés les conditions de test réel et par conséquent ils n'ont pas discuté entre eux pendant les manipulations. Afin de s'assurer d'une certaine qua-lité d'ergonomie, un questionnaire a permis de recueillir les informations pertinentes. Lors de la manipulation, certains

paramètres ont été fixés, comme l'environnement matériel et logiciel (machines mises à disposition), la mission effectuée, le nombre de participants. Cette évaluation est complémentaire de l'évaluation technique du Système Multi-Agents (SMA) seul (aspect technique) et de l'extraction des statistiques de ces manipulations (aspect "apprentissage").

5.3.1 Protocole utilisé

Le protocole pour le sondage effectué est le suivant : les manipulateurs, réunis par groupes de trois et n'ayant jamais manipulé l'application, reçoivent une feuille d'instructions détaillant les étapes de l'expérience. Ils doivent collaborer dans les conditions réelles de manipulation pour effectuer une tâche simple de saisie-dépôt en robot virtuel (pour limiter le biais qu'apporte la manipulation du robot réel sur les performances des manipulateurs). Cette tâche consiste en 4 étapes, chacune associée à un guide : atteinte d'un cylindre, saisie du cylindre, dépôt du cylindre, recul du robot. Le maître de session est le premier manipulateur connecté, aucune contrainte ne lui est indiquée pour céder ce rôle. Pour les mesures de performance, 4 manipulations successives sont demandées, mais le sondage est effectué après les 2 premières manipulations, pour bien mettre en valeur les éventuels problèmes liés à un manque d'ergonomie et d'intuitivité de l'interface.

Le questionnaire est le suivant, les réponses vont de 1 (pas du tout) à 5 (parfaitement), avec possibilité de ne pas se prononcer (N/A) :

- I- Efficacité :
  - I-1- J'ai pu terminer les tâches proposées
  - I-2- J'ai pu terminer rapidement les tâches proposées
  - I-3- J'ai pu terminer efficacement les tâches proposées
  - I-4- Je me suis senti à l'aise en utilisant cette application
  - I-5- Il m'a été facile d'apprendre à utiliser cette application
- II- Efficience
  - II-1- L'aide en ligne est claire
  - II-2- Le graphisme est suffisamment explicite pour me permettre d'accomplir les tâches

  - II-3- Les textes explicatifs sont suffisamment explicites pour me permettre d'accomplir les tâches
  - II-4- La répartition des espaces de l'application me paraît claire
- III- Satisfaction
  - III-1- Cette application m'a plu
  - III-2- Cette application est simple à utiliser
- IV- Navigation
  - IV-1- Les scénarios de manipulation sont explicites
  - IV-2- J'ai pu retrouver les consignes à tout moment
  - IV-3- Les scénarios de manipulation sont efficaces pour accomplir la tâche
  - IV-4- Les scénarios de manipulation sont pertinents pour accomplir la tâche
  - IV-5- Les scénarios de manipulation sont logiques pour accomplir la tâche
- V- Evaluation technique
  - V-1- Le temps d'exécution des actions était-il adéquat ?
  - V-2- Le fonctionnement était-il bon pendant le choix de la mission ?
  - V-3- Le fonctionnement était-il bon pendant le choix des actions ?
  - V-4- Le fonctionnement était-il bon pendant la production ?
  - V-5- Le fonctionnement global était-il bon ?

Une sixième catégorie permet de récupérer les remarques diverses.

5.3.2 Résultats quantitatifs

6 groupes de 3 personnes ont suivi ce protocole. Les groupes sont notés a, b, c, d, e et f, la personne i du groupe x étant notée x_i (ces éléments sont utiles à la compréhention des résultats compilés en annexe D). 3 versions d'interface ont été testées : IHM 1, IHM 2 et IHM 3. Les groupe a et b pour tester l'IHM 1, les groupes c et d ayant bénéficié d'une interface modifiée

(IHM 2) compte tenu des remarques des groupes a et b. Les groupes e et f ont testé l'IHM 3 qui a également bénéficiée des remarques des autres groupes (différences : positionnement des boutons plus logique, ajout de boutons pour faciliter la création des groupes, correction de bugs). Les résultats individuels sont compilés dans des tableaux en annexe D. Une synthèse est faite en tableau 5.1 et 5.2. Le graphique correspondant est donné dans la figure 5.1, qui présente les moyennes par aspect ainsi que leur évolution au fil des versions d'interface.

Catégorie	Moy.	Ecart-Type	Moy. IHM 1	E-T IHM 1
I- Efficacité	4.07	0.54	3.7	0.3
II- Efficience	3.76	0.67	3.63	0.65
III- Satisfaction	4.19	0.6	4.42	0.74
IV- Navigation	4.08	0.54	3.6	0.37
V- Evaluation technique	4.25	0.55	4	0.68
Moyenne	4.07	0.38	3.87	0.41

Tableau 5.1: Synthèse des moyennes (Moy.) et écarts-types (E-T) des notes données, par catégorie, puis comparaison des résultats pour chacune des interfaces.

Catégorie	Moy. IHM 2	E-T IHM 2	Moy. IHM 3	E-T IHM 3
I- Efficacité	4.27	0.37	4.23	0.72
II- Efficience	3.86	0.87	3.79	0.57
III- Satisfaction	4	0.71	4.17	0.26
IV- Navigation	4.19	0.47	4.45	0.21
V- Evaluation technique	4.37	0.5	4.37	0.46
Moyenne	4.14	0.4	4.2	0.31

Tableau 5.2: Suite de la synthèse des moyennes et écarts-types des notes données.

5.3.3 Interprétation des résultats

La première donnée que l'on peut extraire est la moyenne globale des notes données, qui est de 4.07/5, exprimant une qualité d'ergonomie plutôt appréciée. On remarque également que quels que soient les groupes de données traités, l'écart-type est faible (généralement autour de 0.4), indiquant par là qu'il y a peu de divergences de vue entre les utilisateurs.

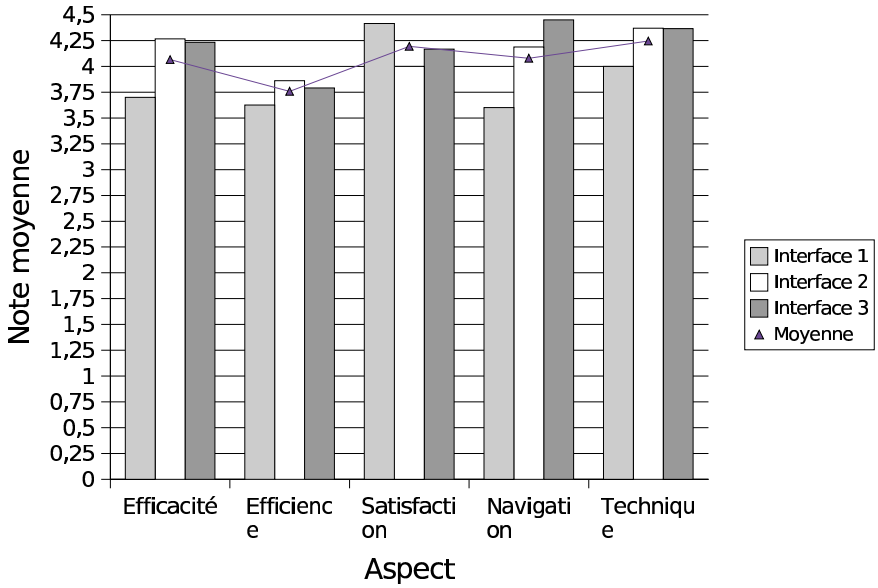

Figure 5.1: La courbe est la moyenne par aspect alors que les histogrammes sont les moyennes pour chaque version d'interface.

Le passage de la moyenne globale de 3.87 à 4.14 puis à 4.2 en changeant quelques points dans l'interface et en corrigeant quelques bugs peut d'ailleurs s'interpréter comme une amélioration de l'ergonomie et du confort, en faisant fi de la petitesse des échantillons. On constate notamment que le changement d'interface a augmenté toutes les moyennes, sauf la satisfaction globale, qui baisse d'à peu près autant qu'augmentent les notes dans les autres catégories bien que le dernier changement d'interface ait fait franchir un grand pas à l'appréciation de la navigation, passant de 4.19 à 4.45. Ceci nous rappelle que les échantillons pris sont de taille trop faible pour pouvoir extraire plus que des grandes tendances.
En comparant les catégories entre elles, on voit que c'est l'efficience qui est le point ayant le plus de marge d'amélioration : le point le plus critiqué dans les commentaires apportés (voir dans la suite) est la séparation de l'interface en deux fenêtres. Bien que les changements d'interface aient amené les informations essentielles sur les deux fenêtres, ce point reste handicapant.
Concernant les notes données par les manipulateurs (moyenne des catégories), elles vont de bonnes à très bonnes, sans qu'aucun manipulateur n'ait été totalement déçu par le test. Les utilisateurs ayant été les plus critiques (a_2, b_1 et c_2 voir D) ont notamment fait des reproches du point de vue de l'efficacité et de l'efficience, ce qui confirme les résultats globalement

observés. Les dernières versions de l'interface ont rendu les utilisateurs moins critiques.

5.3.4 Résultats qualitatifs

Les principales remarques faites dans la catégorie VI sont les suivantes :

- Utilisateur pas assez guidé (corrigé en version 2)
- Guides pas assez explicites (documentation sur le site d'Ariti)
- Manque de clarté de l'interface d'Ariti (corrigé en version 2)
- Manque de souplesse de l'interface (corrigé en version 2)
- Manque d'ergonomie des touches du clavier (corrigé en version 3)
- Affichage graphique trop dense (corrigé en version 3)
- Séparation en deux fenêtres pas pratique
- Non-appréciation des scénarios en Anglais

Ces remarques correspondent bien à l'interprétation quantitative faite précédemment : un manque de clarté au niveau de l'interface est à déplorer, notamment pour guider l'utilisateur dans sa manipulation. La note de 1 donnée par l'utilisateur e_3 (annexe D) à la question "Scénarios explicites" semble s'expliquer par son commentaire : il n'a pas apprécié d'avoir les actions en Anglais.

5.4 Outil d'analyse et d'évaluation de la collaboration

Dans cette section nous présentons brièvement l'outil développé pour l'analyse et l'évaluation de la collaboration. Cet outil bénéficie de l'existence du système multi-agent pour la collaboration (SMA-C) développé et intégré dans le système ARITI-C. En effet, le SMA-C offre une décomposition de la collaboration en trois espaces communication, coordination et production. A chaque utilisateur (client) est associé un agent collaborateur qui gère et assiste la collaboration de l'utilisateur avec les autres utilisateurs du même groupe.

Il s'agit d'une interface développée en PHP qui communique avec une base de données MySQL. Cette base de données est également utilisée par le SMA-C. La figure 5.2 illustre les interactions entre la base de données avec l'interface PHP et le SMA-C.
Cet outil permet également l'administration des différentes données utilisées ou générées par le SMA-C comme les informations sur les utilisateurs, les groupes, les connexions, les missions, les actions, les durées de chaque phase de collaboration, etc.

5.4.1 La base de données de ARITI-C

Dans ce qui suit nous présentons la structure de la base de données utilisée d'une part, par le serveur multi-agent pour la collaboration (SMA-C) et d'autre part par l'interface PHP pour l'administration, l'analyse et l'évaluation de collaboration.

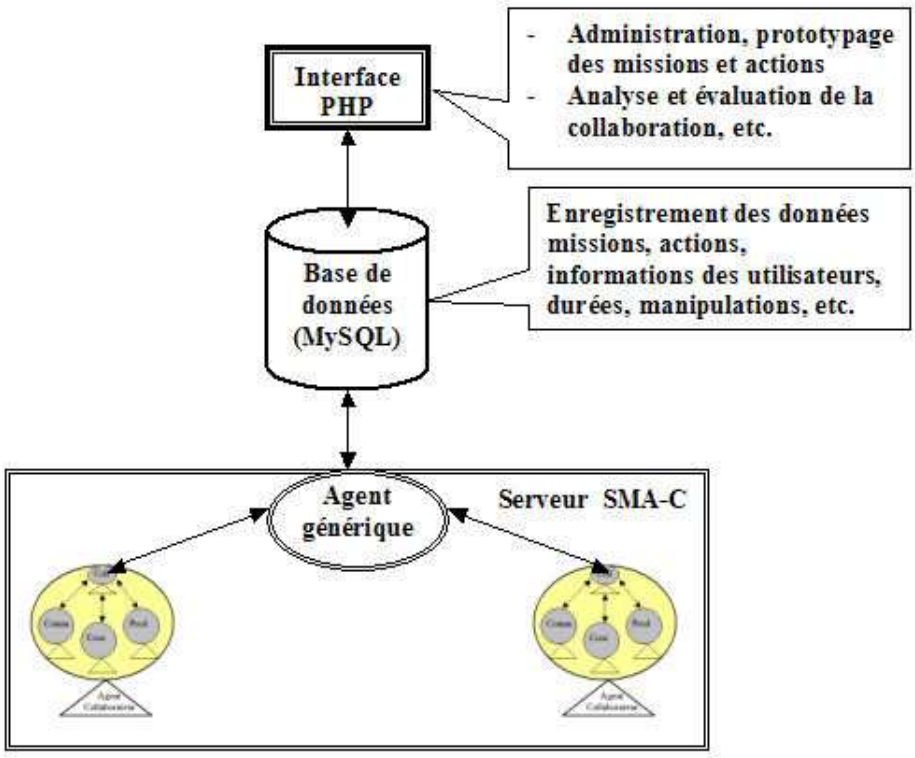

Figure 5.2: L'utilisation de la base de données.

5.4.1.1 Objectifs :

La base de données doit permettre de suivre les actions effectuées par les différentes personnes, la façon dont les groupes ont été constitués, le temps d'action de chaque utilisateur et la performance de leurs manipulations (durée). On veut également évaluer la communication des utilisateurs, en comptant le nombre de messages qu'ils ont échangé. Finalement, on veut également connaître le pays dans lequel chaque personne se trouve pour faire la manipulation, ce qui permet d'analyser la collaboration dans le cas de téléopération à grande distance et entre personnes éloignées les unes des autres.

5.4.1.2 Schéma relationnel et structure de la base de données :

Le schéma relationnel choisi est présenté en figure 5.3. Les connexions de chaque utilisateur sont enregistrées, ainsi que les groupes qu'il a rejoints et les périodes pendant lesquelles il a été maître du groupe (en connaissant les dates d'entrée de chaque utilisateur en mode maître). La décomposition des missions en actions est également stockée dans cette base. A chaque membre du groupe, on peut associer diverses actions effectuées ainsi que la durée de chacune. Les associations `maitre`, `personne_groupe` et `action_effectuee` se traduisent en SQL par des tables, tout comme la relation 1..*/1..* entre `mission` et `action`. Ceci nous donne les tables représentées sur la figure 5.4 (modèle relationnel).

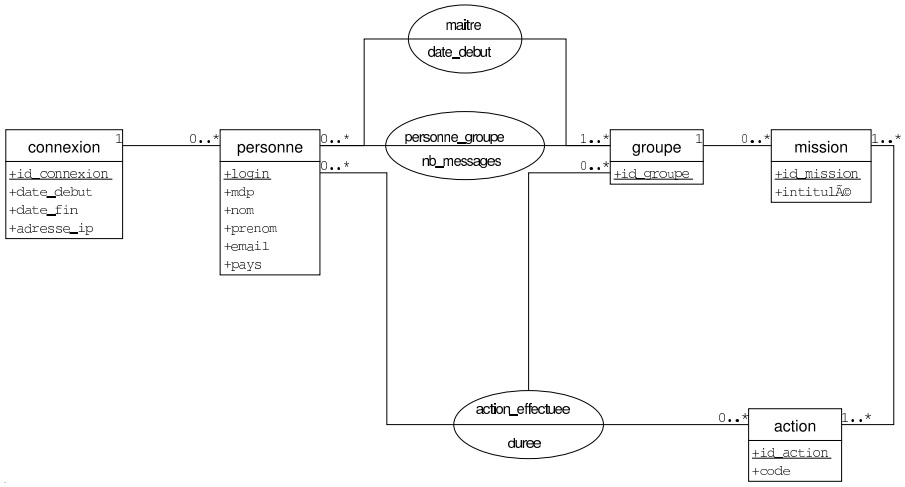

Figure 5.3: Modèle conceptuel de données.

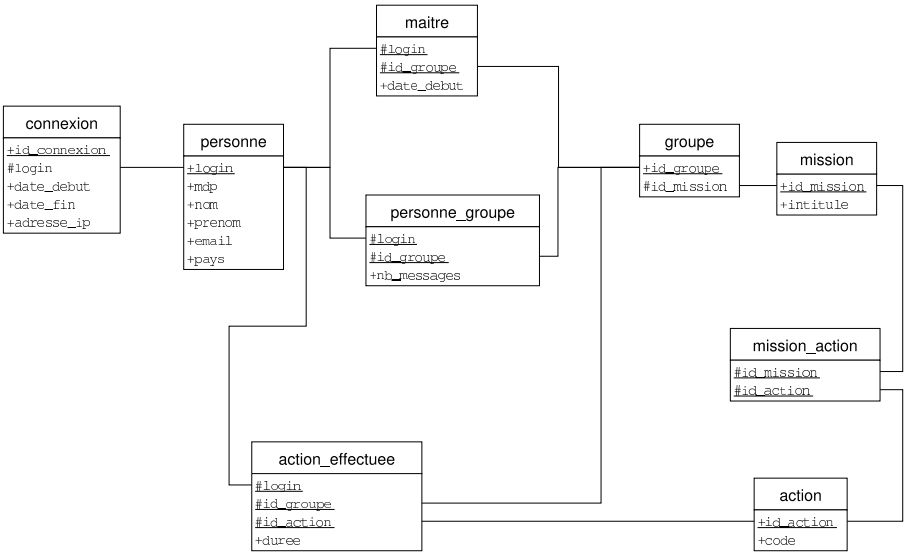

Figure 5.4: Modèle relationnel de données.

Les tables `mission`, `action` et `mission_action` ne pourront pas être modifiées par le serveur multi-agents.

5.4.1.3 Interface d'analyse et d'évaluation de la collaboration

L'interface PHP développée propose de nombreuses possibilités pour l'administration de la base de données et l'analyse de la collaboration. Nous n'allons pas ici détailler toutes ces possibilités, nous nous contenterons seulement de lister les plus importantes :

- Modifier les tables.
- Envoyer un e-mail aux utilisateurs.
- Modifier les missions et les actions.
- Ajouter une mission.
- Re-générer le fichier de calibration de la caméra.
- Purger la base de données.
- Exporter et restaurer la base de données.
- Extraire les statistiques par utilisateur et par groupe.

L'interface PHP et la base de données du système ARITI-C ont été conçues pour pouvoir suivre le fonctionnement du projet, et extraire des caractéristiques relatives aux missions et aux usagers : influence de paramètres de la mission sur les performances des usagers, influence de l'apprentissage pour un même usager ou groupe, influence de la composition du groupe, etc. Ceci permet d'évaluer la façon dont se construit la collaboration pour des tâches complexes et pour présumer de l'intérêt de porter le système sur d'autres types de missions.
Les caractéristiques principalement visées sont :

- L'importance relative des différents espaces de la collaboration (communication, coordination, production). Pour cela, les durées de chaque phase sont mesurées, ainsi que le nombre de messages échangés (faute de pouvoir quantifier *concrètement* le nombre d'informations échangées réellement).

- La façon dont les espaces de collaboration changent d'importance en fonction de l'expérience des usagers. A terme, on suppose que l'essentiel du temps sera passé en production, une fois que chaque usager est habitué aux missions et que chacun sait à quelles tâches il excelle ou lesquelles il préfère. On peut imaginer à terme que le système proposera aux usagers une répartition idéale des tâches en fonction de leur expérience.

- La difficulté d'une mission ou sa complexité de façon quantifiée. On suppose que des missions similaires auront un même profil en termes de durée, de répartition des espaces etc.

- L'influence de l'ergonomie sur les performances. Alors que l'on suit l'évolution des statistiques, on peut déterminer le bon ou le mauvais fonctionnement de certains points voire tester des alternatives, afin de rendre optimal le maniement du logiciel client.

Pour ceci, les données extraites sont les suivantes :

- La durée de chaque espace de la collaboration à chaque fois qu'une mission a été effectuée, ce qui permet de ressortir sa durée moyenne et son écart-type.

- Le nombre de messages échangés au sein du groupe.

- Classées chronologiquement et par groupe, ces données nous permettent de suivre l'évolution du groupe.

- Classées par mission, ces données nous permettent de qualifier la mission.

La figure 5.5 montre une capture d'écran d'une page de l'interface PHP qui permet d'extraire des statistiques par groupe et / ou par mission. Sur cette interface sont affichés la durée de communication, la durée de coordination et la durée de production des différentes actions ainsi que les moyennes et les écarts-types de chaque présentation sous la forme d'histogrammes. Sont également représentés, le tableau des valeurs extraites et l'histogramme des proportions de chaque espace de collaboration.

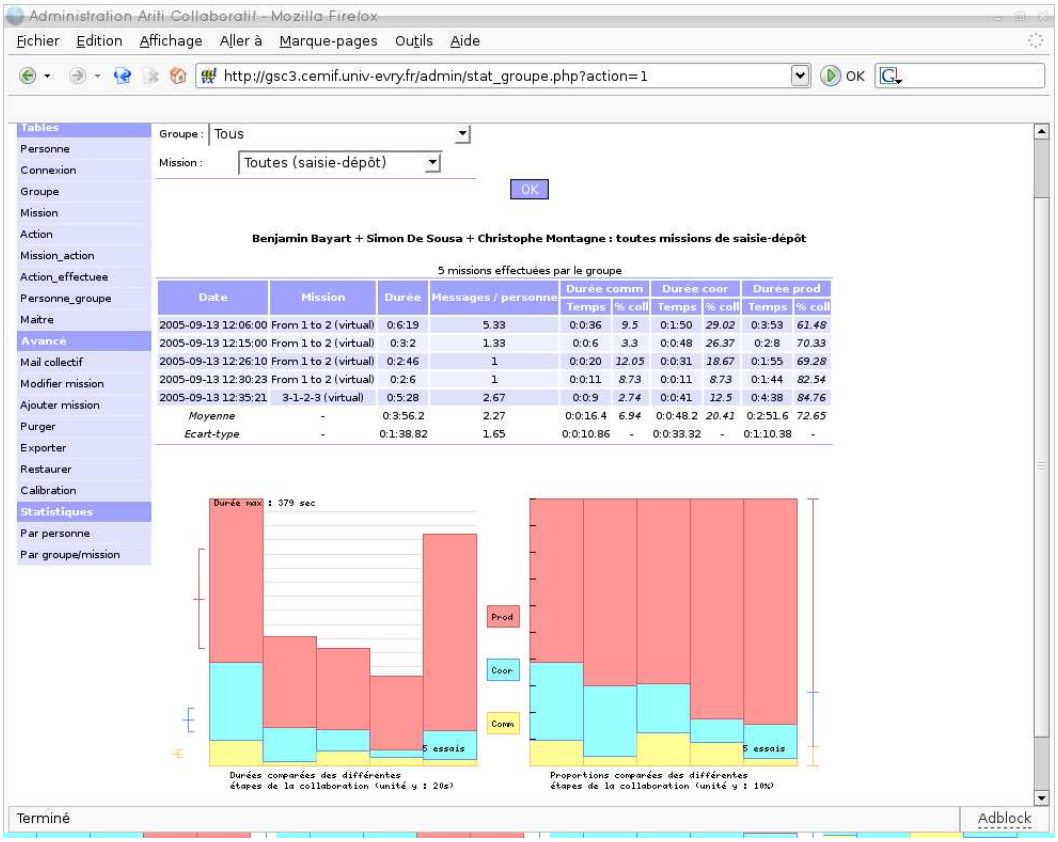

Figure 5.5: Statistiques pour un groupe d'utilisateurs.

Cette partie de l'interface (d'analyse et d'évaluation de la collaboration) est utilisée afin d'analyser et d'évaluer la collaboration des groupes d'utilisateurs utilisant le système ARITI-C. Les statistiques d'utilisation sont présentées et discutées dans la section suivante.

5.5 Statistiques d'utilisation

Dans le même cadre que l'évaluation de l'ergonomie (section 5.3), les statistiques des manipulations des groupes testeurs ont été extraites. Faute d'avoir eu assez de participations extérieures à ce cadre, seules les statistiques de ces manipulations encadrées sont présentées en détail. Pour faciliter l'interprétation des résultats, nous détaillons d'abord le protocole puis ensuite les données chiffrées obtenues. A titre informatif, les résultats généraux de toutes les manipulations sont présentés en dernière partie.

5.5.1 Protocole

Les résultats présentés ont été obtenus d'après le protocole suivant :

- Les groupes sont constitués de 3 personnes n'ayant jamais manipulé Ariti.
- Ces groupes doivent répéter au moins 4 fois la même mission, constituée de 4 actions virtuelles (manipulation du robot virtuel seulement).
- Les conditions d'expérimentation sont aussi réalistes que possible, avec une communication par "chat" uniquement et un minimum de consignes données au préalable.

Dans la pratique, ces expérimentations ont été menées avec 6 groupes. Tous les deux groupes de tests, l'interface a été revue et corrigée et des bugs supprimés.

5.5.2 Résultats

Dans un premier temps, les résultats sont présentés groupe par groupe. En dernière partie les résultats globaux sont exploités. les tableaux de données des statistiques sont donnés en annexe E.

5.5.2.1 Données des groupes 1 et 2

Le groupe 1 a effectuée une série de manipulations "parfaite". Aucun problème ou erreur n'a été à signaler, toutes les données sont donc exploitables telles quelles. Les données sont représentées dans les tableaux (E.1, E.2) de l'annexe E et synthétisées en figure 5.6. Les premières observations sont que, si la première manipulation est plus laborieuse, durant longtemps et faisant intervenir un grand nombre de messages échangés, l'apprentissage

est rapide et le profil de la manipulation s'homogénéise rapidement, dès la deuxième manipulation. Les temps de communication se réduisent alors, pour profiter à la coordination et à la production.

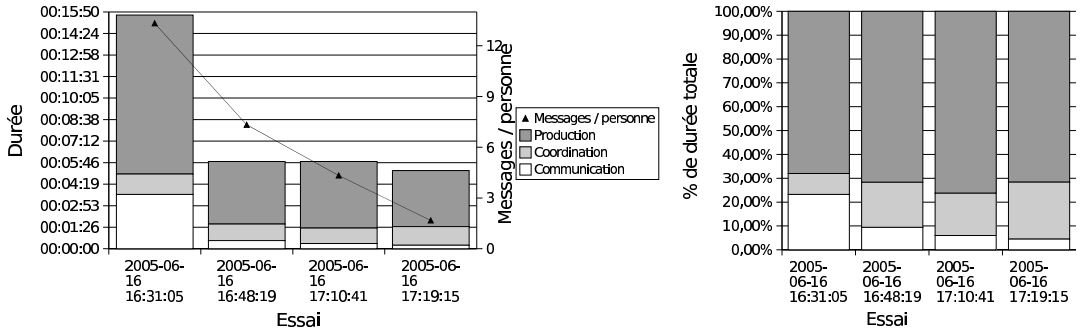

Figure 5.6: Représentation graphique des résultats du groupe 1.

La manipulation du groupe 2 a été plus chaotique que celle du groupe 1 : au troisième essai, une erreur lors du choix de mission a fait arrêter la manipulation en cours (pas de production) et le cinquième essai a fait apparaître un bug qui a bloqué la manipulation. Les résultats sont compilés dans les tableaux (E.3, E.4) de l'annexe E et synthétisées en figure 5.7.

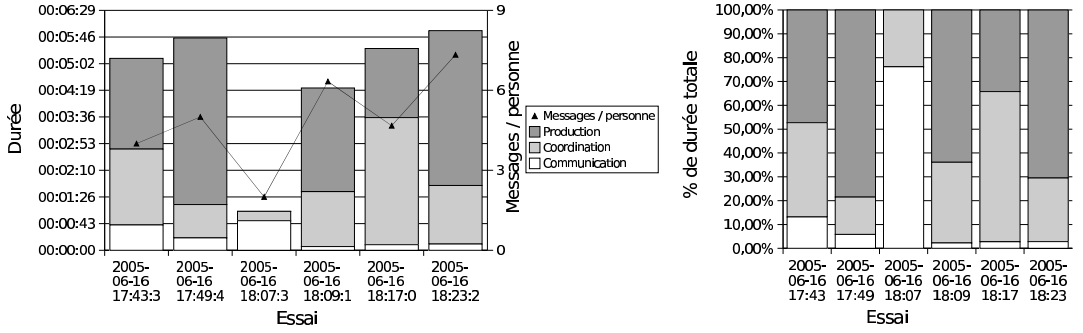

Figure 5.7: Représentation graphique des résultats du groupe 2.

5.5.2.2 Données des groupes 3 et 4

Après passage du groupe 2, certains points de l'ergonomie ont été revus. Les groupes 3 et 4 ont alors fait la manipulation. Comme pour le groupe 2, les manipulations étaient sans fautes. Les premières manipulations sont

plus laborieuses, la phase de communication a duré plus longtemps que les autres phases. Dés les deuxièmes manipulations, nous pouvons remarquer une nette amélioration des différents éléments du groupe (apprentissage efficace). Le groupe 3 les manipulations leurs ont plu alors ils ont voulu rajouter un cinquième essai. Les résultats pour le groupe 3 (respectivement 4) sont présentées dans les tableaux (E.5, E.6) (respectivement E.7, E.8) de l'annexe E. Une synthèse de ces statistiques est donnée dans la figure 5.8 (respectivement 5.9).

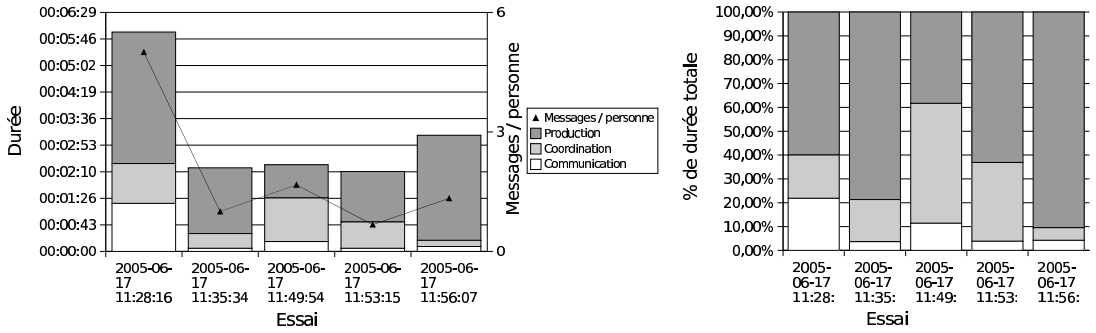

Figure 5.8: Représentation graphique des résultats du groupe 3.

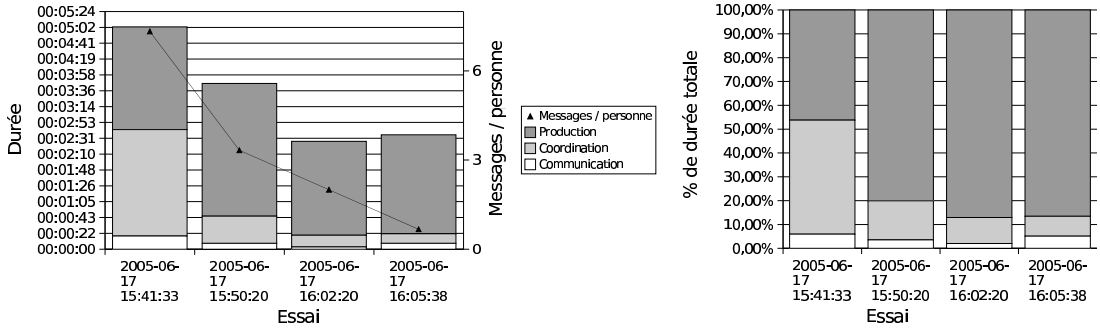

Figure 5.9: Représentation graphique des résultats du groupe 4.

5.5.2.3 Données des groupes 5 et 6

Les tests du groupe 5 sont consécutifs à une refonte importante de l'interface et une correction de bugs majeurs. Ils se sont déroulés sans problème. Les données résultantes sont présentées dans les tableaux (E.9, E.10) de l'annexe

E. La figure 5.10 présente une synthèse de ces résultats. La figure 5.11 montre bien que le groupe 6 ait fait ses missions sans problèmes, sauf que leurs membres étaient beaucoup plus bavards que dans les autres groupes, notamment dans la dernière mission, où le temps de coordination représente bien le nombre de civilités qu'ils se sont échangées. Les données de ce dernier groupe sont présentées dans les tableaux (E.11, E.12) de l'annexe E.

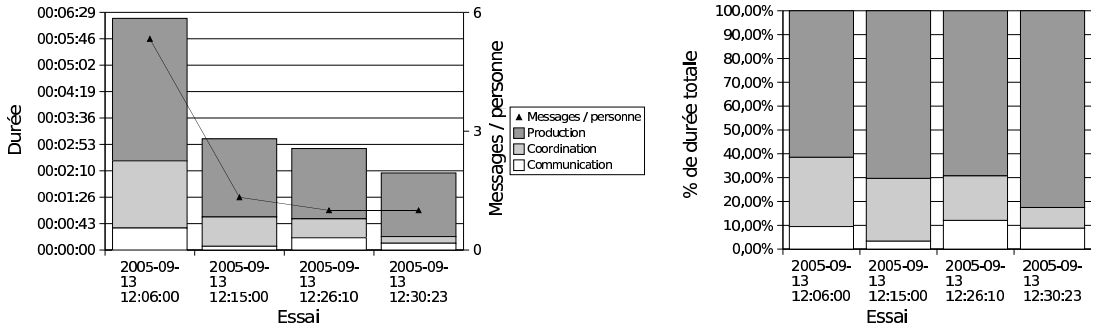

Figure 5.10: Représentation graphique des résultats du groupe 5.

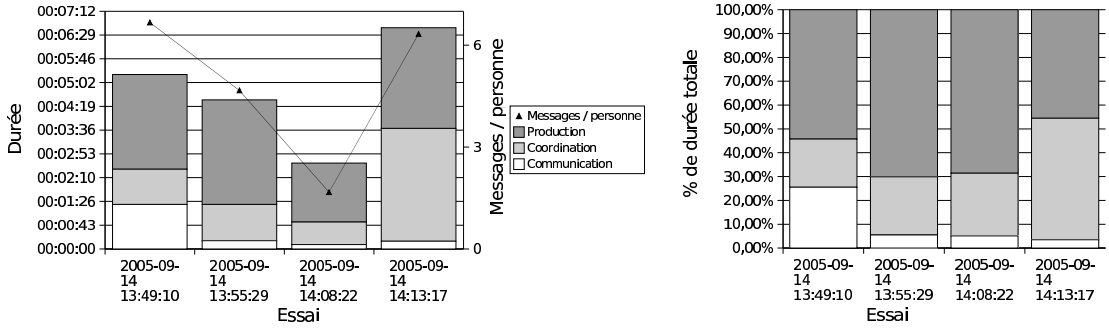

Figure 5.11: Représentation graphique des résultats du groupe 6.

5.5.2.4 Inteprétation

On constate que, malgré des différences de durées entre les groupes, les comportements sont sensiblement les mêmes : la première manipulation est la plus longue, faisant intervenir une plus grande quantité de messages ainsi qu'une durée de communication accrue alors que les missions suivantes, une

fois la prise en main effectuée, se focalisent sur la coordination et la production. L'apprentissage est très rapide, de l'ordre d'une manipulation. Le graphe sur la figure 5.12 montre bien ce phénomène : en classant les barres de l'histogramme dans l'ordre des essais au sein du groupe, on voit une nette tendance à la baisse au fil des essais, se traduisant par une augmentation de la part de coordination et de production (voir le graphe de droite). Pour des raisons de lisibilité, la courbe des nombres de messages n'est pas affichée. Les essais sont classés dans l'ordre suivant : les 6 premières barres représentent les premiers essais de chacun des 6 groupes, les 6 barres suivantes les deuxièmes essais, etc. Ceci permet de mieux faire ressortir le facteur d'apprentissage.

En deuxième observation (voir tableaux (5.3 et 5.4) et figure 5.13), on constate que la répartition des 3 espaces de communication en termes de durée est similaire dans tous les groupes test : autour de 10% du temps pour la communication, de 25% pour la coordination et de 65% pour la production, la partie communication ayant proportionnellement le plus grand écart-type, car c'est elle qui varie essentiellement durant l'apprentissage.
En troisième observation, suite aux modifications relatives à l'ergonomie de l'interface, on constate une baisse de la durée générale de manipulation qui semble indiquer que cette modification a été une amélioration : l'extraction de statistiques nous permet non seulement d'évaluer le groupe mais aussi la qualité du logiciel. En pratique, la partie modifiée de l'interface concernait uniquement les parties communication et coordination : on constate bien sur le graphe de gauche de la figure globale 5.13 que ce sont ces points qui ont bénéficié de ce changement (coordination plus rapide, moins de communication car moins d'ambigüités dans l'interface – après les 10 premières barres de l'histogramme). Pour des raisons de lisibilité, la courbe des nombres de messages n'est pas affichée. Les essais sont classés par ordre chronologique afin de mieux percevoir l'influence des évolutions de l'interface sur l'accomplissement de la mission.

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
Moyenne	*4:42*	*4.04*	*0:27*	*9.7*
Ecart-type	*2:39*	*2.96*	*0:42*	-

Tableau 5.3: Résumé des données extraites pour toutes les manipulations, pour la mission choisie.

Date	Durée coor		Durée prod	
	Temps	%	Temps	%
Moyenne	*1:11*	*25.24*	*3:3*	*65.05*
Ecart-type	*0:51*	-	*1:47*	-

Tableau 5.4: Suite du résumé des données extraites pour toutes les manipulations, pour la mission choisie.

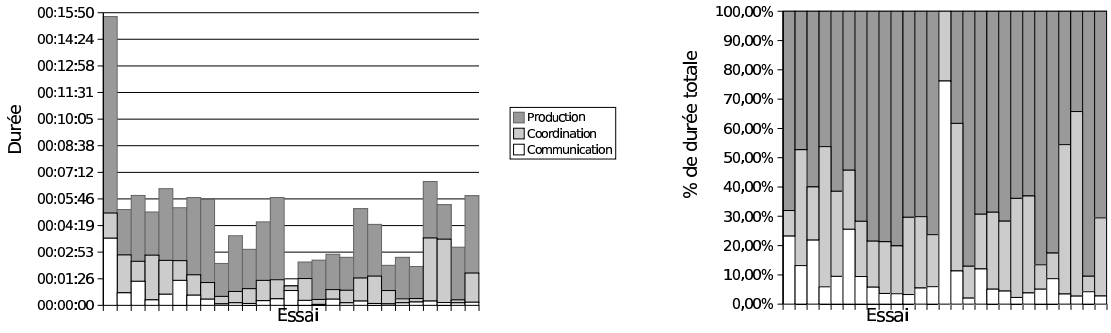

Figure 5.12: Représentation de l'apprentissage pour la mission choisie.

5.5.2.5 Résultats généraux

Cette dernière partie présente les résultats de l'ensemble des manipulations de saisie-dépôt ayant été faites. Les 42 essais sont répartis comme tels :

- 26 essais de mission "From 1 to 2 (virtual)" (4 actions, robot virtuel) réalisés dans le cadre des tests présentés précédemment.
- 3 essais de "From 1 to 2 (virtual)" indépendants des tests.
- 6 essais de "From 1 to 2" (4 actions, robot réel).
- 5 essais de "1 to 2 to 1" (8 actions, robot réel).
- 1 essai de "1 to 2 to 1 (virtual)" (8 actions, robot virtuel).
- 1 essai de "3-1-2-3 (virtual)" (9 actions, robot virtuel).

Les statistiques extraites sont compilées dans les tableaux 5.5 et 5.6, et représentées en figure 5.14. Ces statistiques affectent légèrement la répartition 10% / 25% / 65% des durées observées des différentes étapes de la

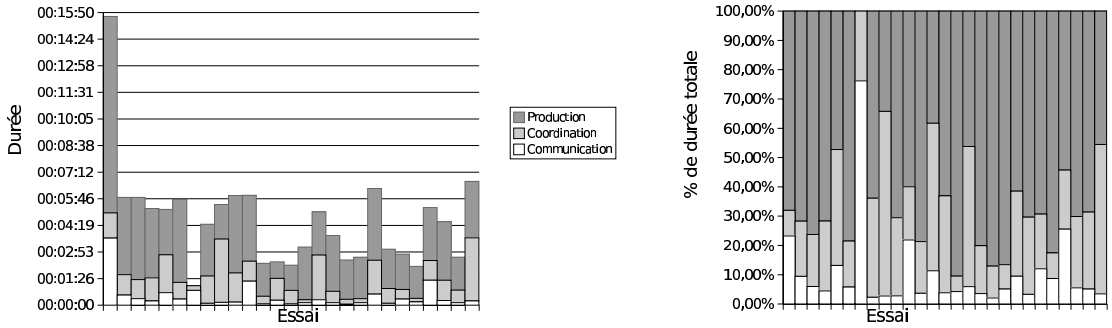

Figure 5.13: Représentation de l'évolution de l'accomplissement de la mission choisie.

collaboration : en situation réelle, la situation semble être plutôt 10% / 30% / 60%. Si les missions qui requièrent le plus d'action sont les plus longues à effectuer, ceci est au moins partiellement compensé par la durée également plus longue de la phase de coordination. On constate également quelques missions avortées, qui n'influencent pas de façon notable les résultats observés dans les conditions "idéales" des tests encadrés.

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
Moyenne	*4:36*	*3.44*	*0:32*	*11.82*
Ecart-type	*2:48*	*4.5*	*0:52*	-

Tableau 5.5: Résumé des données extraites pour toutes les manipulations, pour toutes les missions de saisie-dépôt.

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
Moyenne	*1:22*	*29.85*	*2:41*	*58.33*
Ecart-type	*1:27*	-	*2:2*	-

Tableau 5.6: Suite du résumé des données extraites pour toutes les manipulations, pour toutes les missions de saisie-dépôt.

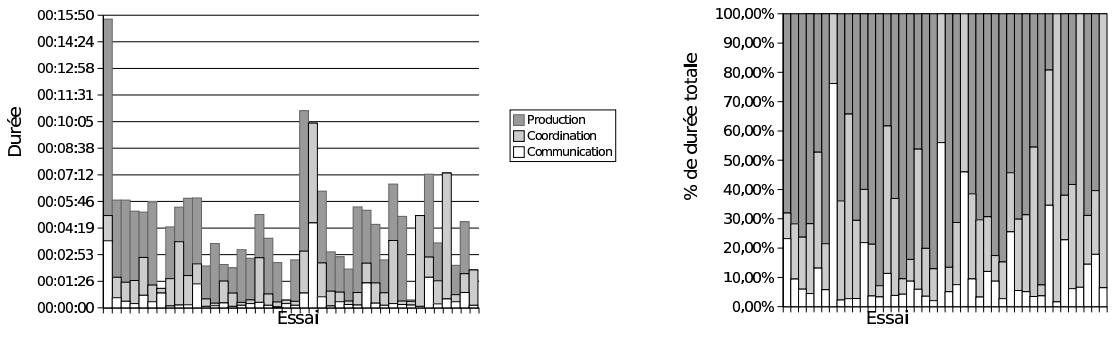

Figure 5.14: Représentation graphique de tous les résultats des missions de saisie-dépôt.

5.6 Bilan

Dans ce chapitre nous avons présenté une évaluation de l'ergonomie de l'interface homme-machine de ARITI-C. Nous avons présenté également l'outil développé pour l'analyse et l'évaluation de la collaboration ainsi que les résultats de cette évaluation.

Nous avons travaillé avec des groupes d'utilisateurs débutants, qui n'ont jamais manipulé le système ARITI auparavent. Ils étaient amenés à exécuter une tâche de collaboration en groupe, ils ont commencé par la communication, ensuite la coordination et enfin ils ont réussi à manipuler le robot en même temps pour que chacun d'entre eux puisse exécuter ses actions.

Les utilisateurs ont répondu à un questionnaire pour évaluer l'ergonomie de l'interface homme-machine de ARITI-C. Nous avons par conséquent amélioré l'interface d'ARITI-C, notamment en classant les boutons par type, en supprimant les fonctions obsolètes et en explicitant plus clairement les fonctions des boutons. Nous avons également amélioré la souplesse de l'interface collaborative (fenêtres redimensionnables, signaux "clignotants" lorsqu'une information importante est à noter). En dernier nous avons amélioré l'intuitivité par l'utilisation systématique d'info-bulles pour les différents éléments d'interface.
Nous avons présenté l'outil développé pour analyser et évaluer la collaboration. Ce dernier est utilisé pour extraire des statistiques d'utilisation. Les résultats se montrent encourageants, le système est operationnel et les résultats montrent bien que tous les agents du système fonctionnent proportionnellement selon la tâche en-cours d'exécution.
Au final, nous pouvons prétendre que notre but, de permettre à plusieurs utilisateurs de collaborer pour réaliser des missions de téléopération du robot est atteint.

Conclusion et Perspectives

Conclusion générale

Tout au long de cette thèse, nous avons essayé d'atteindre notre objectif principal qui consiste à proposer un système pour la collaboration (collecticiel) destiné à la téléopération collaborative via Internet. L'objectif de cette recherche a été de modéliser, de concevoir, d'implémenter et d'évaluer un système de collaboration. Ce dernier doit être capable de prendre en charge la collaboration de plusieurs utilisateurs distants pour les aider à préparer et à réaliser une mission commune de téléopération d'un robot. Dans ce manuscrit, nous avons vu que les systèmes de téléopération existants, ne permettent pas une réelle collaboration des utilisateurs pour réaliser une mission. Il n'y a pas de communication et de coordination des utilisateurs pour manipuler le robot. Ces systèmes ne supportent pas la collaboration des utilisateurs faute du manque de collecticiels dans le domaine de la téléopération. Nous avons pallié à ces manques en proposant une solution basée sur un formalisme multi-agent pour la téléopération collaborative via Internet.

Le premier chapitre nous a permis d'identifier les concepts, les méthodes et les outils nécessaires pour répondre à notre problématique en étudiant les deux domaines de recherche qui sont les SMA et le TCAO. Dans le domaine des SMA, le formalisme d'agent proposé par Ferber ainsi que la plateforme de développement JADE ont été retenus et utilisés. Dans le domaine du TCAO, nous avons retenu le concept du trèfle fonctionnel des collecticiels pour la modélisation de la collaboration ainsi que le principe du modèle d'architecture PAC* pour la conception de l'architecture logicielle du collecticiel.

Dans le second chapitre, nous avons présenté la modélisation d'un Système Multi-Agent pour la Collaboration (SMA-C). Nous avons proposé un formalisme pour la collaboration en se basant sur le concept du trèfle fonctionnel

des collecticiels identifié dans le premier chapitre. Nous avons également proposé un formalisme d'Agent Collaborateur (AC) en se basant d'une part, sur le formalisme d'agent de Ferber et d'autre part, sur le formalisme de collaboration proposé. Le SMA-C est composé de plusieurs ACs, et chaque AC correspond à un client connecté. L'architecture logicielle du SMA-C est inspirée de l'architecture logicielle des collecticiels PAC*. Elle comprend un ensemble de trois agents chacun correspond à une dimension du trèfle fonctionnel : Agent communication, Agent coordination et Agent production. Les ACs sont utilisés pour réduire les interactions laborieuses entre utilisateurs, donc maximiser l'efficacité de la communication. Ils sont créés dans le but d'aider les utilisateurs à mieux collaborer.

Dans le troisième chapitre nous avons présenté la conception, l'implémentation et l'évaluation du SMA-C proposé. Le langage UML a été utilisé pour la conception de l'architecture logicielle du SMA-C et la plateforme de développement JADE a été utilisée pour l'implémentation de cette architecture logicielle. Nous avons également proposé un simulateur pour l'évaluation du SMA-C. Nous avons démontré que le SMA-C est un système stable pour la collaboration durant toutes ses phases : La communication, la coordination et la production, bien que la production ne dépende pas directement de notre SMA-C mais plutôt du type du système à manipuler et du travail à produire.

Le quatrième chapitre nous a permis de tester le SMA-C (développé et validé en simulation) sur une application réelle avec des données et des contraintes réelles. Nous l'avons appliqué sur le système de téléopération mono-utilisateur ARITI afin de le rendre collaboratif. Ce système étant utilisé depuis 2000 par des centaines d'utilisateurs dans le monde et que le besoin de le rendre collaboratif se faisait sentir. Nous avons implémenté une interface de collaboration qui fait le lien entre le système ARITI, le SMA-C et les utilisateurs. Nous avons présenté la nouvelle Interface Homme-Machine (IHM) de ARITI-C ainsi que l'IHM de collaboration avec les différentes missions pouvant être utilisées en collaboration. Une première version de ce système est mise sur le site WEB du laboratoire depuis juillet 2005 afin d'évaluer l'ergonomie de l'IHM de ARITI-C et d'analyser la collaboration des utilisateurs.

Dans le dernier chapitre, nous avons présenté une évaluation de l'ergonomie de l'IHM de ARITI-C ainsi que les statistiques d'utilisation obtenues grâce

à l'outil mis en oeuvre pour l'analyse et l'évaluation de la collaboration. Les tests effectués avec des utilisateurs montrent que le nouveau système de téléopération collaborative ARITI-C est un système stable. Il permet aux utilisateurs de communiquer, de coordonner (choisir leurs actions), de produire (exécuter leurs tâches), en utilisant ou pas l'assistance des guides virtuels ou encore en déléguant l'exécution de ces tâches à leurs agents pour les exécuter à leurs places.

En résumé, nous avons proposé un système multi-agent dédié à la collaboration basé sur les propriétés d'agents et sur les caractéristiques d'un TCAO. Les apports de notre contribution sont liés à l'application envisagée à savoir la téléopération collaborative. Nous pouvons ainsi prétendre que notre système de téléopération ARITI-C supporte le travail collaboratif. Sur le plan utilisabilité, ARITI-C assiste et supervise les utilisateurs pendant le déroulement des missions, il permet également d'analyser et d'évaluer la collaboration au sein du groupe. D'un point de vue conceptuel, l'architecture logicielle proposée est ouverte et peut être utilisé dans d'autres domaines. Cela dit, notre modèle reste néanmoins limité pour son manque de souplesse. Dans le cas d'activités collaboratives plus générales notre architecture reste inadaptée et c'est pour cette raison que nous envisageons d'améliorer le coté fonctionnel en modifiant les fonctionnalités de coordination. Nous envisageons aussi d'offrir une grande marge d'initiative et plus de liberté aux utilisateurs (le choix du maître par exemple) durant leurs manipulations. Nous prévoyons également d'étendre les fonctionnalités de collaboration pour contrôler la caméra en même niveau que le robot.

Perspectives

Le système ARITI-C, est actuellement en phase de test et d'évaluation à grande échelle par des utilisateurs se trouvant dans les quatre coins du monde. L'analyse des remarques et des statistiques obtenues nous permettra de l'améliorer et de le rendre plus efficace.

L'interface PHP et la base de données du système ARITI-C ont été conçues pour pouvoir suivre le fonctionnement du projet, et extraire des caractéristiques relatives aux missions et aux usagers : influence des paramètres de la mission sur les performances des usagers, influence de l'apprentissage

pour un même usager ou groupe, influence de la composition du groupe, etc. Ceci permet d'évaluer la façon dont se construit la collaboration pour des tâches complexes et pour présumer de l'intérêt de porter le système sur d'autres types de missions et d'applications dans le futur.

La façon dont les espaces de collaboration changent d'importance en fonction de l'expérience des usagers est une information importante, on peut imaginer à terme que le système proposera aux usagers une répartition idéale des tâches en fonction de leur expérience.

Un autre travail envisagé est d'exploiter le SMA-C pour le télétravail collaboratif autour des environnements de Réalité Virtuelle (RV) et de Réalité Augmentée (RA). Il s'agit d'adapter le SMA-C pour la conception et le développement de nouvelles architectures logicielles de collaboration adaptées aux nouvelles IHM multisensorielles (nouvelles modalités de perception d'interaction et de communication) offertes par les environnements de RV/RA.

Une première application porte sur le développement d'un collecticiel pour la téléopération collaborative de robot semi-immersif à travers une visualisation de grande taille stéréoscopique (virtuelle et/ou réelle augmentée) permettant à deux opérateurs distants de percevoir l'environnement tridimensionnel du robot. Par ailleurs, un retour haptique permettant aux deux opérateurs d'avoir aussi une perception kinesthésique des opérations sera disponible. Cette application s'inscrit dans le cadre du télétravail collaboratif en réalité augmentée et virtuelle autour des deux plateformes EVR@[1] (illustrée en figure 15) et PREVISE[2].

[1] http://lsc.univ-evry.fr/techno/EVRA/index.html
[2] http://www.istia.univ-angers.fr/LISA/FICHES/previse.pdf

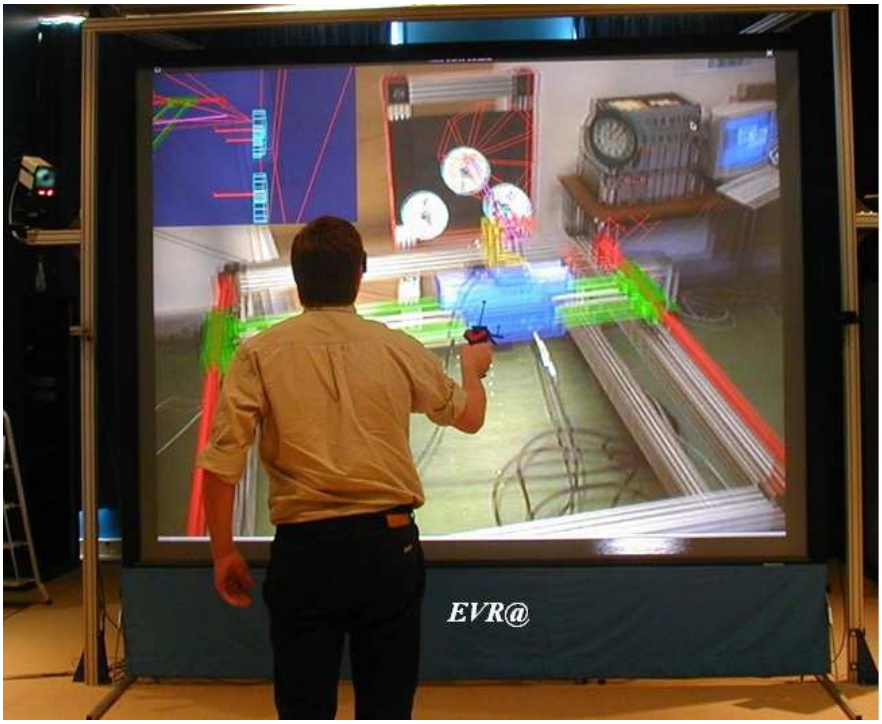

Figure 15: La plateforme de télétravail en réalité virtuelle et augmentée du LSC

Références bibliographiques

[ABD 04] Abdoessalam A. M. et Mehandjiev N., Collaborative Negotiation in Web Service Procurement, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 47-52, 2004.

[AGR 87] Agre P. et Chapman D., Pengi : An Implementation of a Theory of Activity, *AAAI 87 Seattle*, 1, 1987.

[ALA 96] Alaoui A., Laferriere T. et Meloche D., Apprendre En Collaboration Avec D'autres le Travail En Équipe. Théorie et Pratique À L'intention Des Étudiants et Des Étudiantes Du Premier Cycle, *Faculté des sciences de Laval : université de Laval. http://www.fse.ulaval.ca/fac/tact/fr/html/sites/guide2.html1*, 1996, consulté sur la Toile en février 2000.

[ARL 04] Arlabosse F., Gleizes M. P. et Occello M., *Méthodes de Conception*, Dans Systémes Multi-Agents, 2004.

[BAS 92] Bass L., Faneuf R., Little R., Mayer N., Pellegrino B., Reed S., Seacord R., Sheppard S. et Szczur. M., A Metamodel for the Runtime Architecture of an Interactive System, *Journal, ACM Special Interest Group Computer-Human Interface bulletin (SIGCHI)*, 24, 1, 32-37, 1992.

[BEL 96] Bellamine N., Contributions Méthodologiques á la Conception de Collecticiels : Application Aux Réunions de Planification Coopérative Du Satellite Multi-Instrument SOHO, *Thèse de doctorat en informatique*, 1996, Université Toulouse 1.

[BOO 92] Booch G., *Conception Orientée Objets et Applications*, Addison-Wesley, 1992.

[BOW 95] Bowers J. et Button G., Workflow from Within and Without, *Actes de la conférence European Conference on Computer Supporter Cooperative Work (ECSCW 95)*, 51-66, 1995, Kluwer Academic.

[BRA 97] BRADSHAW J. M., Software Agents, *Cambridge MA: AAAI Press/MIT Press*, 1997.

[BRA 98] BRAVE S., ISHII H. et DAHLEY A., Tangible Interfaces for Remote Collaboration and Communication, *ACM Computer Supported Cooperative Work (CSCW 98)*, 169-178, 1998.

[BRA 99] BRAZIER F. M., JONKER C. M. et TREUR J., Compositional Design and Reuse of a Geniric Agent Model., *Proceeding of Knowledge Acquisition Workshop (KAW 99)*, 1999.

[BRO 91] BROOKS R., Intelligence Without Reason, *Artificial Intelligence*, 1-27, 1991.

[BUR 92] BURMEISTER B. et SUNDERMEYER K., Cooperation Problem-Solving Guided by Intentions and Perception, WERNER E. ET DEMAZEAU Y. E., , *Decentralized AI*, Elsevier Science Publishers, 1992.

[BUX 92] BUXTON W., Telepresence : Integrating Shared Task and Person Spaces, *In Baecker 93*, 816-822, 1992.

[CAB 03] CABRI G., FERRARI L. et LEONARDI L., A Case Study in Role-Based Agent Interactions, *12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 03)*, 2003.

[CAB 04] CABRI G., FERRARI L. et LEONARDI L., Towards the Use of Mobile Agent Based Message Systems, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 27-32, 2004.

[CAL 97] CALVARY G., COUTAZ J. et NIGAY L., From Single-User Architectural Design to PAC*: A Generic Software Architecture Model for CSCW, *Proceedings of the ACM CHI 97*, 242-249, Mars 1997.

[CAR 99] CARRE P. et CASPAR P., Traité Des Sciences et Des Techniques de la Formation, 1999, Paris : Dunod.

[CAS 00] CASTERAN J., GLEIZES M. et GLIZE P., Des Méthodologies Orientées Multi-Agent, *Journées Francophones En Intelligence Artificielle Distribuée et Systèmes Multi-Agents (JFIADSMA 00)*, 191-207, Septembre 2000.

[COH 88] COHEN P. et LEVESQUE H., Rational Interaction as the Basis for Communication, *SRI Technical Note 433*, 1988.

[COL 92] COLEMAN D. et SHAPIRO R., Defining Groupware, *Special Adversiting Section to Network World*, Juin 1992.

[COU 94] COUTAZ J. et HERMANN M., Adèle et le Médiateur-Compositeur ou Comment rendre une Application Interactive indépendante de l4Interface Usager, *Actes du deuxième colloque de Génie Logiciel (AFCET 84)*, 195- 212, 1994.

[COU 99] COUTAZ J., BÉRARD F., CARRAUX E. et ASTIER W., CoMedi : Using Computer Vision to Support Awareness and Privacy in Mediaspaces, *ACM Conference on Human Factors and Computing Systems (CHI 99)*, 13-14, 1999, ACM press.

[DAV 01] DAVID B. T., IHM pour les collecticiels, *Réseaux et Systèmes Répartis*, 169-206, Novembre 2001, Hermès, Vol. 13.

[DEL 01] DELOACH S. A., Analysis and Design Using MaSE and AgentTool, *12 Th Midwest A.I. and Cognitive Science Conference (MAICS 01)*, 2001, Ohaio.

[DEM 91] DEMAZEAU Y. et MÜLLER J., Decentralized Artificial Intelligence, *Esleiver Science*, 1991.

[DEM 01] DEMAZEAU Y., VOYELLES, Habilitation À Diriger Des Recherches, *INP de Grenoble*, 2001.

[DEW 99] DEWAN P., *Architectures for Collaborative Applications*, Computer-Supported Cooperative Work (Beaudouin-Lafon éditeur), Trends in Software, 1999.

[DEW 00] DEWAN P., Techniques for Evaluating Collaboration Toolkits, *Proceedings of the 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 92-97, 2000.

[DOU 92] DOURISH P. et BLY S., Portholes: Supporting Awareness in a Distributed Work Group., *ACM Conference on Human Factors and Computing Systems (CHI 92)*, 541-547, 1992.

[DOU 96] DOURISH P., Consistency Guarantees : Exploiting Application Semantics Consistency Management in a Collaboration Toolkit, *ACM Computer Supported Cooperative Work (CSCW 96)*, 268-277, 1996, ACM Press.

[DUR 89] DURFEE E., *Coordination of Distributed Problem Solvers*, Kluwer Academic Publishers, 1989.

[ELL 91] ELLIS C., GIBBS S., SIMON et REIN G., Groupware: Some Issues and Experiences, *Journal, Communications of the ACM (CACM)*, 34, 1, 38-58, 1991.

[ELL 94] ELLIS C. A. et WAINER J., A Conceptual Model of Groupware, *ACM Conference on Computer Supported Cooperative Work (CSCW 94)*, 79-88, 1994, Chapel-Hill, North Carolina, USA.

[ELL 99] ELLIS C., *Workflow Technology*, Computer-Supported Cooperative Work (Beaudouin-Lafon éditeur), Trends in Software, 1999, pages 29-54.

[FAR 98] FARATIN P., SIERRA C. et JENNINGS N., Negotiation Decision Functions for Autonomous Agents, *Robotics and Autonomous Systems*, 24, 3-4, 159- 182, 1998.

[FER 95] FERBER J., Les Systèmes Multi-Agents: Vers Une Intelligence Collective, *InterEditions, Paris*, 1995.

[FER 96] FERBER J. et MULLER J., Influences and Reaction: a Model of Situated Multiagent Systems, *ICMAS 96, Kyoto*, December 1996.

[FER 97] FERBER J., Les Systèmes Multi-Agents: Un Aperçu Général, *Revue Techniques et Sciences Informatiques*, 16, 8, 1997.

[FLO 88] FLORES F., GRAVES M., HARTFIELD B. et WINOGRAD T., Computer Systems and the Design of Organizational Interaction, *Journal, Transactions on Information Systems (ToIS)*, 6, 2, 153-172, 1988.

[FRE 03] FREY D., STOCKHEIM T., WOELK P. et ZIMMERMANN R., Integrated Multi-Agent-Based Supply Chain Management, *12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 03)*, 2003.

[GOL 95] GOLDBERG K., MASCHA M., GETNER S. et ROTHENBERG N., Desktop Teleoperation Via the World Wide Web, *IEEE Conference on Robotics and Automation*, 1995, Nagoya, Japan.

[GOL 00] GOLDBERG K., CHEN B., SOLOMON R., BUI S., FARZIN B., HEITLER J., POON D. et SMITH G., Collaborative Teleoperation Via the Internet., *IEEE International Conference on Robotics and Automation*, 24-27 Avril 2000, San Francisco Californie.

[GOL 01] GOLDBERG K., The Tele-Actor: A New Framework for Collaborative Telepresence., *Conference on Computer Human Interaction CHI*, 31 Mars- 5 Avril 2001, Seattle, Washington.

[GOL 02] GOLDBERG K., SONG D., KHOR Y., PESCOVITZ D., LEVANDOWSKI A., HIMMELSTEIN J., SHIH J., HO A. et PAULOS E., Collaborative Online Teleoperation with Spatial Dynamic Voting and

a Human "Tele-Actor", *IEEE International Conference on Robotics and Automation*, 11-15 Mai 2002, Arlingthon VA , Washingthon DC.

[GRA 97] GRAHAM N. et URNES T., Integrating Support for Temporal Media Into an Architecture for Graphical User Interfaces, *International Conference on Software Engineering (ICSE 97)*, 172-182, 1997.

[GRE 85] GREEN M., Report on Dialogue Specification Tools, *Journal, User Interface Management Systems (UIMS), Eurographics Seminars*, 9-20, 1985, Springer-Verlag.

[GRU 94] GRUDIN J., CSCW : History and Focus, *Jounal, IEEE Computer*, 27, 5, 19-26, 1994.

[GUE 99] GUESSOUM Z. et BRIOT J., From Active Objects to Autonomous Agents, *IEEE Concurrency*, 7, 3, 68-79, November 1999.

[GUI 04] GUIDI-POLANCO F., CUBILLOS C. et MENGA G., The Agent-Based GAP: A Framework for Global Automation Systems, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 53-58, 2004.

[GUL 98] GULKNECHT O. et FERBER J., A Meta-Model for the Analysis and Design of Organizations in Multi-Agents Systems, *Proceedings of ICMAS 98, IEEE Computer Society*, 1998.

[GUT 00] GUTWIN C. et GREENBERG S., The Mechanics of Collaboration: Developing Low Cost Usability Evaluation Methods for Shared Workspaces, *in Proceedings of the IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 00)*, 2000.

[HAL 00] HALL T., Practioner's Guide to Evaluating Collaboration Systems, *The MITRE Corporation, July 2000, Available at: http://collaboration.mitre.org/practguide/PractionersGuide.htm*, 2000.

[HAY 79] HAYES-ROTH B., A Cognitive Model of Planning, *Cognitive Science*, 3, 1979.

[HAY 85] HAYES-ROTH B., A Blackboard Architecture for Control, *Artificial Intelligence*, 1985.

[HER 04] HEROUX D. P., Illuminating Data Sources as Founts of Agent Based Collaboration, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 59-64, 2004.

[HIL 94] HILL R., BRINK T., ROHAL S., PATTERSON J. et WAYNE W., The RendezVous Architecture and Language for Constructing Multi-User Applications., *ACM Transactions on Computer Human Interaction (ToCHI)*, 1, 2, 81-125, 1994.

[ISH 94] ISHII H., KOBAYASHI M. et ARITA K., Iterative Design of Seamless Collaborative Media, *Journal Communication of the ACM*, 37, 8, 83-97, 1994, ACM Press.

[JAC 99] JACOBSON J., BOOCH G. et RUMBAUGH J., The Unified Software Development Process, 1999.

[JEN 98] JENNINGS N. R. et (ED) M. J. W., Agent Technology : Foundations, Applications and Markets, *Springer-Verlag*, 1998.

[KAR 94] KARSENTY A., Le Collecticiel: De L'interaction Homme-Machine . La Communication Homme-Machine-Homme., *Technique et Science Informatiques,*, 13, 1, 105-127, 1994.

[KHA 04] KHALIL-IBRAHIM I., KOTSIS G. et KRONSTEINER R., Substitution Rules for the Verification of Norm-Compliance in Electronic Institutions, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 21-26, 2004.

[KHE 04] KHEZAMI N., OTMANE S. et MALLEM M., A Formal Model for Collaborative Teleoperation, *the International Conference on Computing, Communications and Control Technologies (CCCT 04)*, 183-187, 2004, August 14-17, 2004, Texas, USA.

[KHE 05a] KHEZAMI N., OTMANE S. et MALLEM M., An Approach to Modelling Collaborative Teleoperation, *12th IEEE International Conference on Advanced Robotics (ICAR 05)*, 2005, July 18-20, Seattle, Washington, USA.

[KHE 05b] KHEZAMI N., OTMANE S. et MALLEM M., Modelling and Evaluation of a Multi-Agent System for a Collaboration, *International Federation of Automatic Control (PRAHA IFAC World Congress 2005)*, 2005, July 3-8, PRAHA, Czech Republic.

[KHE 05c] KHEZAMI N., OTMANE S. et MALLEM M., A new Formal Model of Collaboration by Multi-Agent Systems, *IEEE 2005 International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS 05)*, 32-37, 2005, April 18-21, Waltham, Massachusetts, USA.

[KHE 05d] KHEZAMI N., OTMANE S. et MALLEM M., A New Interface for Collaborative Teleoperation, *International Federation of Automatic Control (PRAHA IFAC World Congress 2005)*, 2005, July 3-8, PRAHA, Czech Republic.

[KIR 03] KIRN S., HEINE C., HERRLER R. et KREMPELS K., Agent.Hospital -Agent-Based Open Framework for Clinical Applications, *12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 03)*, 2003.

[KNU 00] KNUTILLA A., STEVES M. et ALLEN R., Workshop on Evaluating Collaborative Enterprises-Workshop Report, *Proceedings of the 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises WETICE 00*, 79-85, 2000.

[KRA 88] KRASNER G. et POPE S., A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80., *Journal of Object Oriented Programming (JOOP)*, 1, 3, 26-49, 1988.

[LAB 98] LABBANI-IGBIDA O., MULLER J. et BOURJAULT A., An Emergentist Methodology to Design and Evaluate Collective Bahaviours in Robot's Colonies, *Collective Robotics*, LNCS/LNAI 1456, 72-84, 1998.

[LAU 02] LAURILLAU Y., Conception et Réalisation Logicielles Pour Les Collecticiels Centrées sur Lactivité de Groupe : Le Modèle et la Plate-Forme Clover, *Thése de doctorat en informatique*, 2002, Université Joseph Fourier - Grenoble I.

[LEN 75] LENAT D., BEINGS : Knowledge an Interacting Experts, *International Joint Conference on Artificial Intelligence*, 1975.

[LEO 96] LEONNEC J., Etude de cas: le projet Pyramide, *8e forum France Telecom Recherche, France Telecom, Paris*, 1996.

[LEP 04] LEPARC P., VAREILLE J. et MARCÉ L., E-productique ou contrôle et supervision distante de systèmes mécaniques sur l'internet, *JESA (Journal européen des systèmes automatisés) numéro 5, Volume 38*, 525-558, May 2004.

[LIN 01] LIND J., Iterative Software Engineering for Multiagent Systems: The MASSIVE Method, *LNCS*, 1994, 2001.

[LOP 99] LOPRIORE L., L'apprentissage Coopératif : Un Défi Pour Les

Professeurs de Langue, *Le français dans le monde recherches et applications No. spécial apprendre les langues étrangères autrement*, 134-141, 1999.

[MAA 03] MAAMAR Z., AKHTER F. et LAHKIM M., An Agent-Based Approach to Specify a Web Service-Oriented Environment, *12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 03)*, 2003.

[MAC 99] MACKAY W., *Media Spaces: Environments for Informal Multimedia Interaction*, Computer-Supported Cooperative Work (Beaudouin- Lafon éditeur), Trends in Software, 1999.

[MOU 96] MOULIN B. et BRASSARD M., A Scenario Based Design and an Environment for Multiagent Systems, *Distibuted Artificial Intelligence : Architecture and Modelling, Proceedings of First Australian Workshop on DAI*, LNAI 1087, 216-232, 1996.

[MUL 98] MULLER J., Vers Une Méthodologie de Conception de Systèmes Multiagents de Résolution de Problèmes Par Émergence, *Actes des Sixièmes Journées Francophones d'Inteligence Artificielle Distribuée et Systèmes Multi-Agents*, 1998.

[NIG 94] NIGAY L., Conception et Modélisation Logicielles Des Systèmes Interactifs : Application Aux Interfaces Multimodales, *Thèse de doctorat Informatique*, 1994, Université Joseph Fourier, Grenoble, France.

[NWA 96] NWANA H. S., Software Agents: An Overview, *Knowledge Engineering Review*, 11, 3, 1996.

[OTM 00a] OTMANE S., MALLEM M., KHEDDAR A. et CHAVAND F., Active virtual guide as an apparatus for augmented reality based telemanipulation system on the internet, *IEEE Computer Society - 33rd Annual Simulation Symposium (ANSS 00)*, 2000, pages 185–191, Wyndham City Center Hotel, Washington, D.C., USA.

[OTM 00b] OTMANE S., MALLEM M., KHEDDAR A. et CHAVAND F., ARITI : an Augmented Reality Interface for Teleoperation on the Internet., *Advanced Simulation Technologies Conference (ASTC 00) - High Performance Computing*, 2000, pages 254–261, Wyndham City Center Hotel, Washington, D.C., USA.

[OTM 00c] OTMANE S., Télétravail Robotisé et Réalité Augmentée : Application à la Téléopération via Internet, *Thèse de doctorat en robotique*, 2000, Université d'Evry Val-Essonne.

[OUA 94] OUADOU K., AMF : Un Modèle Darchitecture Multi-Agents Multifacettes Pour Interfaces Homme-Machine et Les Outils Associés, *Thèse de doctorat Informatique*, 1994, Ecole Centrale de Lyon.

[PAL 02] PALEN L. et GRUDIN J., *Discretionary Adoption of Group Support Software: Lessons from Calendar Applications*, Organizational implementation of collaboration technology (Munkvold éditeur), 2002.

[PAT 94] PATTERSON J., A Taxonomy of Architectures for Synchronous Groupware Applications. Actes Du Groupe de Travail, *Workshop on Software Architectures for Cooperative Systems : ACM Conference on Computer-Supported Cooperative Work (CSCW 94)*, 317-328, 1994.

[PIC 04] PICARD G. et GLEIZES M. P., The Adelfe Methodology to Appear in Methodology for Enginneering for Agent Systems, *Bergenti F., Gleizes M-P., Zambonelli F. Editors Kluwer*, 2004.

[PIN 00] PINELLE D. et GUTWIN C., A Review of Groupware Evaluations, *IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises,*, 86-91, 2000.

[RIC 00] RICORDEL P. M. et DEMAZEAU Y., From Analysis to Deployment: A MultiAgent Platform Survey, *International Workshop on Engineering societies in the Agents World (ECAI 00)*, 2000.

[SAL 95] SALBER D., De L'interaction Individuelle Aux Systèmes Multiutilisateurs. L'exemple de la Communication Homme-Homme- Médiatisée, *Thèse de doctorat Informatique*, 1995, Université Joseph Fourier, Grenoble.

[SAU 98] SAUCY P. et MONDALA F., KhepOnTheWeb: One year of Access to a Mobile Robot on the Internet, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 98)*, Victoria, B.C. Canada, 23-30, Oct. 12-17, 1998.

[SCH 98] SCHULZ D., BURGARD W. et CREMERS A., Predictive Simulation of Autonomous Robots for Tele-Operation Systems Using the World Wide Web, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 98)*, Victoria, B.C. Canada, 31-36, Oct. 12-17, 1998.

[SEL 04] SELLIAH S., REDDY R., YU J., BHARADWAJ V. et REDDY S., Eksarva: A Framework for Enabling Agent-Based Collaboration, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 33-38, 2004.

[SHO 93] SHOHAM Y., Agent Oriented Programming, *Artificial Intelligence*, 60, 1993.

[SHO 97] SHOHAM Y., An Overview of Agent-Oriented Programming, *In Software Agents, J. M. Bradshaw (Ed.)*, 271-290, 1997.

[SIM 98] SIMMONS R., Xavier: An Autonomous Mobile Robot on the Web, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 98)*, Victoria, B.C. Canada, 43-48, Oct. 12-17, 1998.

[SIR 00] SIRE S., La Collaboration Directe : Un Paradigme Dínteraction Pour Le Travail Collaboratif Assisté Par Ordinateur, *Thèse de doctorat en informatique*, 2000, UNIVERSITE TOULOUSE 1.

[STE 93] STEINER D., BURT A., KOLB M. et LERIN C., The Conceptual Framework of Mail, *Maamaw 93*, August 1993, Neuchâtel, Switzerland.

[STE 98] STEIN M., Painting on the World Wode Web: The PumaPaint Project, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 98)*, Victoria, B.C. Canada, 37-42, Oct. 12-17, 1998.

[STE 04] STEFANO A. D. et SANTORO C., Designing Collaborative Agents with EXAT, *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 04)*, 15-20, 2004.

[SYC 98] SYCARA K., Multiagent Systems, *AI Magazine*, 19, Juillet-Août 1998.

[TAR 97] TARPIN-BERNARD F., Travail Coopératif Synchrone Assisté Par Ordinateur : Approche AMF-C, *Thèse de doctorat Informatique*, 1997, Ecole Centrale de Lyon.

[TAY 95] TAYLOR K. et TREVELYAN J., Australia's Telerobot On The Web, 26^th *International Symposium On Idustrial Robots*, Singapore, Oct. 1995.

[WEI 99] WEISS G., *Multiagent Systems A Modern Approach to Distributed Artificial Intelligence*, The MIT Press, 1999.

[WOO 95] WOOLDRIDGE M. et JENNINGS N., Intelligent Agent : Theory and Practice, *Knowledge Engineering Review*, 10, 2, 1995.

[WOO 01] WOOLDRIDGE M. et CIANCARINI P., Agent-Oriented Software Engineering: The State of the Art, *M. Wooldridge and P. Ciancarini, ed. AOSE 01*, 2001.

Annexes

Annexe A

FIPA : un standard pour agents et systèmes multi-agents

La FIPA "Foundation for Intelligent Physical Agents" a été fondée en 1996 .C'est une organisation à but non lucratif dont l'objectif est de produire des standards logiciels pour des agents hétérogènes et des systèmes Multi -Agents (SMA) portés par des plateformes différentes. Le but est de rassembler les dernières avancées dans la recherche sur les SMA et les pratiques industrielles dans le logiciel, les réseaux et les systèmes d'information.
L'accent a été mis sur les utilisations commerciales et industrielles des SMA. Le but est de rassembler les dernières avancées dans la recherche sur les SMA et les pratiques industrielles dans le logiciel,les réseaux et les systèmes d'information.
La mission que s'est fixée la FIPA est de faciliter l'interopérabilité des agents et des systèmes multi-agents provenant de différents fournisseurs. Ainsi, la FIPA a produit un ensemble de spécifications qui s'étendent des langages de communications (Agent Communication Languages) aux langages de contenu (Content Language) ainsi qu'aux protocoles d'interaction. Le leitmotiv de la FIPA est que l'utilisation des actes de langage, de la logique des prédicats et de la définition d'ontologies permet de garantir une interprétation non ambiguë, respectant la sémantique des messages échangés.
La figure A.1 illustre l'approche de la FIPA en ce qui concerne la plateforme, qui se compose de trois composants principaux : la couche de communication (Message Transport System), la gestion des agents (Agent Management System) et la gestion de l'annuaire (Directory Facilitator). Ce sont ces composants qui sont évalués lors des tests d'interopérabilité.
La spécification du transport de messages entre agents FIPA gère la livraison et la représentation des messages à travers des protocoles de transport de réseau différents. Au niveau du transport, un message consiste en une

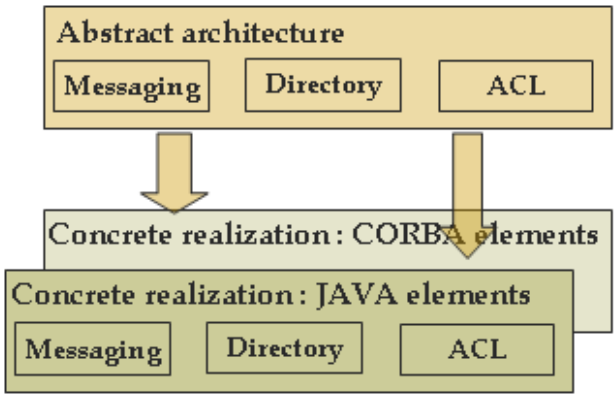

Figure A.1: Architecture abstraite de la plateforme FIPA

enveloppe et un corps.La figure A.2 détaille la structure d'un message FIPA ainsi que l'envoi d'un message entre deux agents situés sur des plateformes distinctes. Le message est construit dans un premier temps par l'agent A, avec le corps du message représentant la sémantique de l'échange et l'enveloppe qui regroupe les informations de transport (encodage, protocole, ...). L'agent A délègue ensuite l'expédition du message au MTS (le service de transport de message) qui en fonction du protocole utilisé (IIOP, HTTP, RMI, ...) sélectionne un Message Transport Provider (MTP) et effectue la communication avec la plateforme hébergeant l'agent B.
La spécification de gestion des agents FIPA fournit la structure dans laquelle les agents existent et fonctionnent. Elle établit le modèle de référence logique pour la création, l'enregistrement, la localisation, la communication, la migration et le retrait d'agents. Elle décrit les primitives et les ontologies nécessaires pour supporter les services suivants dans une plateforme d'agents:

- Les pages blanches : la localisation, la désignation des agents et le contrôle de l'accès aux services sont fournis par l'Agent Management System (AMS). Les noms des agents sont représentés par une structure flexible et extensible appelée identificateur d'agent, qui peut supporter des noms sociaux, des adresses de transport, des services de résolution de nom.

- Les pages jaunes: la localisation des services et l'enregistrement des services sont fournis par le Directory Facilitator (DF).

- Les services de transport de messages.

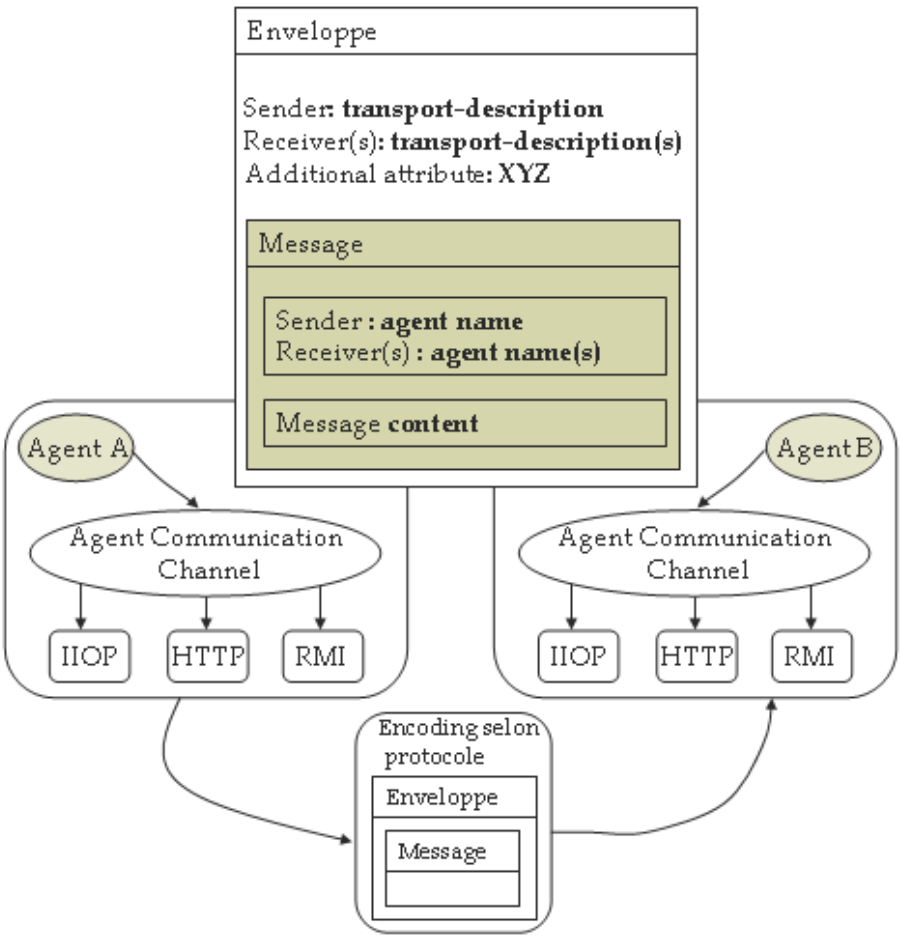

Figure A.2: Envoi de message entre deux agents FIPA

La notion de plateforme est utilisée comme une infrastructure de communication et de gestion des agents. Ces fonctionnalités sont apportées au travers de services de plateformes accessibles soit aux agents de la plateforme, soit aux autres plateformes.

Annexe B

Interface de télécalibration

B.1 Présentation

Il s'agit de de développer un module pour la calibration à distance de la caméra. Ce module permet de récupérer la matrice de transformation définissant la projection des points 3D sur l'affichage 2D.

Le principe de fonctionnement est le suivant : une vue en haute résolution (640*480) de la scène est proposée, ainsi que le placement de 13 points de cette scène. L'utilisateur peut sélectionner des points et les déplacer pour ajuster le positionnement. Il peut également choisir de ne pas tenir compte de certains points lors du calcul (points non visibles). Un clic sur le bouton de calibration permet de calculer la calibration relative à ces points sur la vue 640*480. Ce résultat est stocké localement en mémoire de l'applet. Si le positionnement convient et que l'utilisateur rentre le bon mot de passe, en cliquant sur le bouton de mise à jour du serveur, l'utilisateur demande de calculer les paramètres de calibration pour la vue en basse résolution (320*240) de l'interface principale. Ces paramètres sont envoyés par protocole http à un script php sur le serveur. Ce script génère un fichier qui sera chargé par tous les clients d'ARITI-C lancés par la suite. Cette solution est très simple et pratique.
Une capture de l'interface obtenue est donnée en figure B.1.

B.2 Architecture logicielle

Les 3 classes principales de cette fonctionnalité sont regroupées dans le package `calibration` (figure B.2). Elles font toutefois appel à d'autres classes du package `ariti`, pour l'affichage de la vidéo et le calcul des projections d'objets 3D.

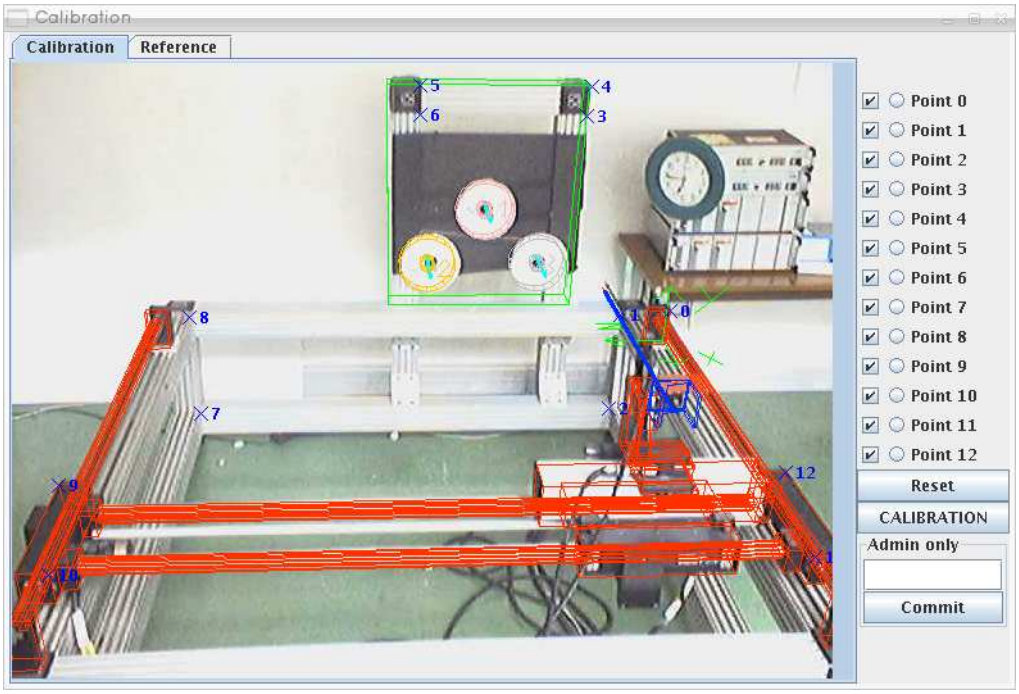

Figure B.1: Interface de calibration.

- `FrameCalibration` : il s'agit de la classe de la fenêtre de calibration, avec son interface.
- `VueCalibration` : c'est la classe qui superpose l'image vidéo, la scène virtuelle et les points de calibration. Elle est instanciée par `FrameCalibration`.
- `CalibreCamera` : cette classe effectue les calculs de calibration.

Le fonctionnement des classes `FrameCalibration` et `VueCalibration` est trivial. Le calcul de calibration de `CalibreCamera` a été repris tel quel de l'ancienne version d'Ariti, dans laquelle il n'était pas documenté.

B.3 Côté serveur

Du côté du serveur, un fichier `calib.txt` contient sur chaque ligne un paramètre de la matrice de projection en vue 320*240 (11 paramètres).

Un script `calibration.php` récupère les 11 paramètres http et régénère `calib.txt` si tous les paramètres sont définis. Par sécurité, l'interface web

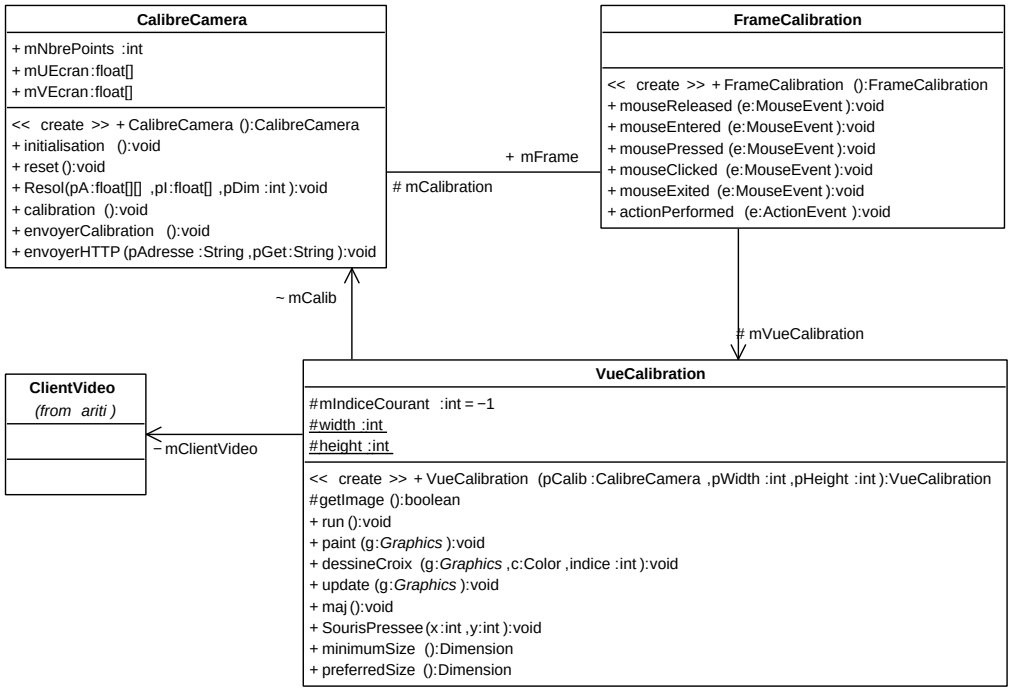

Figure B.2: Diagramme de classes de la calibration.

de maintenance / extraction de données d'ARITI-C, offre un script permettant de régénerer le fichier de configuration aux valeurs par défaut, qui sont donc stockées en dur dans le script.

Annexe C

Interface de création de guides virtuels

L'interface de création de guides est constituée de 3 vues virtuelles correspondant à celles de l'interface principale, et d'un panneau de contrôle permettant de créer et paramétrer les guides. Les différents contrôles offerts sont :

- Créer guide : ouvre la fenêtre de création de guide (voir plus loin). Le guide nouvellement créé écrase le guide couramment sélectionné.
- Nouveau : désélectionne le guide courant pour pouvoir insérer un nouveau guide sans en écraser un autre.
- Supprimer : supprime le guide courant.
- Lier deux objets : permet de lier un guide au guide courant.
- Délier objets : supprime le lien entre les guides sélectionnés.
- Déformer l'objet : ouvre une interface de déformation de guide. Pour déformer un guide, il suffit de tirer un de ses points par un glissement de souris dans une des vues.
- Déplacer l'objet (bouton) : ouvre une fenêtre de saisie de coordonnées de translation pour l'objet.
- Sélectionner un objet : permet de sélectionner un objet d'un clic. Sa fenêtre de propriétés (voir plus loin) apparaît alors.
- Axe de rotation : affiche l'axe de rotation actuellement sélectionné pour l'objet.

- Touner l'objet : permet de tourner l'objet d'un glissement de souris dans une des vues. L'axe et l'angle de la rotation sont définis dans le panel intitulé comme tel.

- Déplacer l'objet (bouton radio) : permet de translater l'objet en glissant la souris dans une des vues.

- Valider : en mode de création de guide simple, ouvre une fenêtre permettant d'insérer le guide dans l'environnement. En mode de création collaborative de guide, permet de valider la création courante et d'envoyer l'état de la création de guide à tous les collaborateurs.

- Récupérer : permet de récupérer dans l'environnement de création de guides le dernier guide inséré dans l'environnement.

- Charger guide : ouvre une interface permettant de récupérer un guide sauvegardé sur le serveur connaissant son nom.

Les deux autres fenêtres principales de cette interface sont la fenêtre de création et paramétrage des guides et la fenêtre de propriétés de guide. La première fenêtre permet de choisir le type de guide et les paramètres de l'équation le définissant et la seconde fenêtre permet de changer le nom du guide, de l'enregistrer sur le serveur (en fournissant le bon mot de passe) et d'en connaître ou modifier toutes les propriétés.

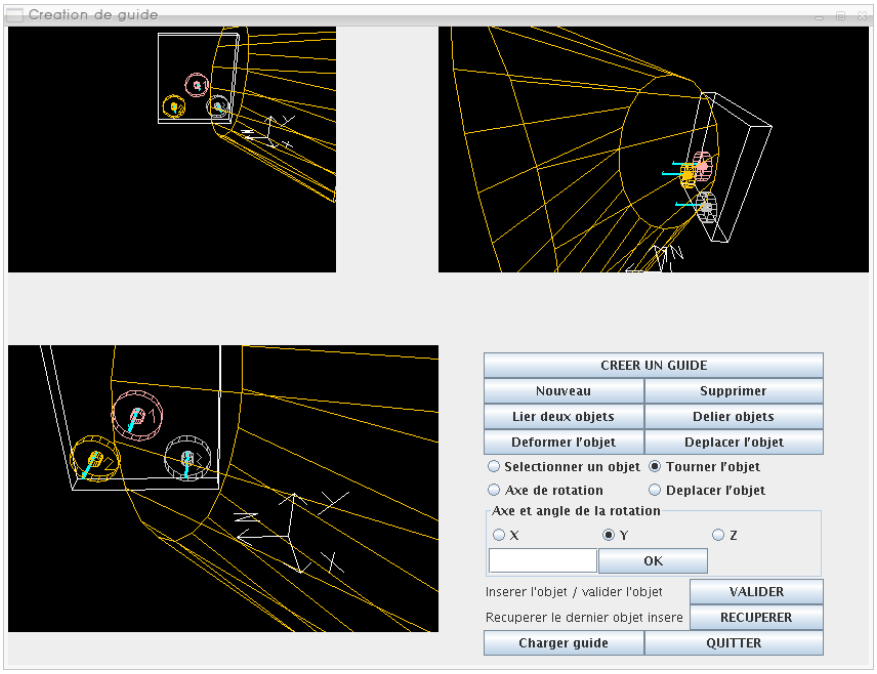

Figure C.1: Interface de création des guides

Annexe D

Données extraites du questionnaire d'évaluation de l'ergonomie de l'interface homme-machine d'ARITI-C

Les résultats individuels sont compilés dans les tableaux D.1 et D.2 pour l'efficacité, D.3 et D.4 pour l'efficience, D.5 et D.6 pour la satisfaction, D.7 et D.8 pour la navigation, dans D.9 et D.10 pour l'aspect technique.

Question	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
I-1	3	3	5	4	5	5	5	5	5
I-2	2	3	4	4	3	3	4	5	5
I-3	4	4	4	3	5	4	4	5	5
I-4	5	3	3	3	4	2	4	1	5
I-5	5	3	4	4	3	4	4	3	4
Moyenne	3.8	3.2	4	3.6	4	3.6	4.2	3.8	4.8
Ecart-type	1.3	0.45	0.71	0.55	1	1.14	0.45	1.79	0.45

Tableau D.1: Résultats quantitatifs obtenus pour la catégorie de questions I : Efficacité.

Question	d_1	d_2	d_3	e_1	e_2	e_3	f_1	f_2	f_3
I-1	5	5	4	5	5	5	5	4	5
I-2	5	4	4	4	5	5	4	2	5
I-3	5	4	5	5	5	5	4	2	5
I-4	4	4	3	4	3	5	4	3	3
I-5	4	4	4	5	4	4	5	3	4
Moyenne	4.6	4.2	4	4.6	4.4	4.8	4.4	2.8	4.4
Ecart-type	0.55	0.45	0.71	0.55	0.89	0.45	0.55	0.84	0.89

Tableau D.2: Suite des résultats quantitatifs obtenus pour la catégorie de questions I : Efficacité.

Question	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
II-1	4	3	5	3	4	-	3	-	5
II-2	4	4	3	3	5	2	5	4	5
II-3	3	3	4	3	5	3	4	4	5
II-4	4	4	3	-	5	4	4	1	4
Moyenne	3.75	3.5	3.75	3	4.75	3	4	3	4.75
Ecart-type	0.5	0.58	0.96	0	0.5	1	0.82	1.73	0.5

Tableau D.3: Résultats quantitatifs obtenus pour la catégorie de questions II : Efficience.

Question	d_1	d_2	d_3	e_1	e_2	e_3	f_1	f_2	f_3
II-1	5	4	-	5	-	-	4	4	-
II-2	4	4	2	5	3	4	2	3	4
II-3	5	4	3	4	4	4	4	3	5
II-4	5	4	3	4	3	3	3	4	-
Moyenne	4.75	4	2.67	4.5	3.33	3.67	3.25	3.5	4.5
Ecart-type	0.5	0	0.58	0.58	0.58	0.58	0.96	0.58	0.71

Tableau D.4: Suite des résultats quantitatifs obtenus pour la catégorie de questions II : Efficience.

Question	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
III-1	5	3	4	5	5	5	4	3	5
III-2	5	3	5	4	5	4	4	3	5
Moyenne	5	3	4.5	4.5	5	4.5	4	3	5
Ecart-type	0	0	0.71	0.71	0	0.71	0	0	0

Tableau D.5: Résultats quantitatifs obtenus pour la catégorie de questions III : Satisfaction.

Question	d_1	d_2	d_3	e_1	e_2	e_3	f_1	f_2	f_3
III-1	5	3	4	5	4	5	4	4	4
III-2	4	4	4	4	4	4	4	4	4
Moyenne	4.5	3.5	4	4.5	4	4.5	4	4	4
Ecart-type	0.71	0.71	0	0.71	0	0.71	0	0	0

Tableau D.6: Suite des résultats quantitatifs obtenus pour la catégorie de questions III : Satisfaction.

Question	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
IV-1	4	3	4	4	3	4	4	5	5
IV-2	3	4	3	4	3	4	3	5	5
IV-3	2	-	4	2	3	4	-	-	4
IV-4	4	4	3	3	5	4	4	-	4
IV-5	4	4	4	3	5	4	4	-	4
IV-6	4	3	3	2	5	4	4	-	5
Moyenne	3.5	3.6	3.5	3	4	4	3.8	5	4.5
Ecart-type	0.84	0.55	0.55	0.89	1.1	0	0.45	0	0.55

Tableau D.7: Résultats quantitatifs obtenus pour la catégorie de questions IV : Navigation.

Question	d_1	d_2	d_3	e_1	e_2	e_3	f_1	f_2	f_3
IV-1	4	3	3	4	5	1	5	4	-
IV-2	-	4	2	5	5	-	-	4	5
IV-3	4	4	4	5	-	-	-	3	-
IV-4	4	4	5	4	4	5	5	5	-
IV-5	-	4	5	4	5	5	4	4	-
IV-6	-	4	5	-	5	5	4	4	5
Moyenne	4	3.83	4	4.4	4.8	4	4.5	4	5
Ecart-type	0	0.41	1.26	0.55	0.45	2	0.58	0.63	0

Tableau D.8: Suite des résultats quantitatifs obtenus pour la catégorie de questions IV : Navigation.

Question	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
V-1	2	3	4	3	3	5	4	-	5
V-2	4	4	4	3	5	5	4	-	5
V-3	4	4	4	3	5	5	4	5	4
V-4	3	3	5	3	5	5	4	3	4
V-5	5	3	5	4	5	4	4	3	5
Moyenne	3.6	3.4	4.4	3.2	4.6	4.8	4	3.67	4.6
Ecart-type	1.14	0.55	0.55	0.45	0.89	0.45	0	1.15	0.55

Tableau D.9: Résultats quantitatifs obtenus pour la catégorie de questions V : Technique.

Question	d_1	d_2	d_3	e_1	e_2	e_3	f_1	f_2	f_3
V-1	-	5	5	4	4	3	4	5	-
V-2	5	4	5	4	3	4	5	4	5
V-3	5	4	5	5	4	4	5	4	5
V-4	5	4	5	4	4	5	5	4	5
V-5	4	4	5	5	4	4	5	4	5
Moyenne	4.75	4.2	5	4.4	3.8	4	4.8	4.2	5
Ecart-type	0.5	0.45	0	0.55	0.45	0.71	0.45	0.45	0

Tableau D.10: Suite des résultats quantitatifs obtenus pour la catégorie de questions V : Technique.

Annexe E

Données des statistiques d'utilisation de ARITI-C

E.1 Données des groupes 1 et 2

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
2005-06-16 16:31:05	0:15:38	13.33	0:3:38	23.24
2005-06-16 16:48:19	0:5:50	7.33	0:0:33	9.43
2005-06-16 17:10:41	0:5:50	4.33	0:0:21	6
2005-06-16 17:19:15	0:5:14	1.67	0:0:14	4.46
Moyenne	0:8:8	6.67	0:1:11.5	14.65
Ecart-type	0:4:20.22	4.34	0:1:24.85	-

Tableau E.1: Données récupérées après les 4 manipulations du premier groupe.

Dans ce tableau "Mess./Pers." représente le nombre de messages échangés par personne au sein du groupe, il permet d'évaluer la communication. Pour des raisons de lisibilité, les durées de chaque action ne sont pas représentées, mais elles sont exploitées en réalité (idem pour les tabelaux E.3, E.5, E.7, E.9 et E.11).

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
2005-06-16 16:31:05	0:1:22	8.74	0:10:38	68.02
2005-06-16 16:48:19	0:1:6	18.86	0:4:11	71.71
2005-06-16 17:10:41	0:1:2	17.71	0:4:27	76.29
2005-06-16 17:19:15	0:1:15	23.89	0:3:45	71.66
Moyenne	0:8:8	14.6	0:5:45.25	70.75
Ecart-type	0:4:20.22	-	0:2:49.68	-

Tableau E.2: Suite des données récupérées après les 4 manipulations du premier groupe.

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
2005-06-16 17:43:38	0:5:11	4	0:0:41	13.18
2005-06-16 17:49:44	0:5:44	5	0:0:20	5.81
2005-06-16 18:07:36	0:1:3	2	0:0:48	76.19
2005-06-16 18:09:17	0:4:23	6.33	0:0:6	2.28
2005-06-16 18:17:03	0:5:27	4.67	0:0:9	2.75
2005-06-16 18:23:29	0:5:56	7.33	0:0:10	2.81
Moyenne	*0:4:37.33*	*4.89*	*0:0:22.33*	*8.05*
Ecart-type	*0:1:40.31*	*1.7*	*0:0:16.38*	-

Tableau E.3: Données récupérées après les 6 manipulations du groupe 2.

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
2005-06-16 17:43:38	0:2:3	39.55	0:2:27	47.27
2005-06-16 17:49:44	0:0:54	15.7	0:4:30	78.49
2005-06-16 18:07:36	0:0:15	23.81	0:0:0	0
2005-06-16 18:09:17	0:1:29	33.84	0:2:48	63.88
2005-06-16 18:17:03	0:3:26	63	0:1:52	34.25
2005-06-16 18:23:29	0:1:35	26.69	0:4:11	70.51
Moyenne	*0:4:37.33*	*34.98*	*0:2:38*	*56.97*
Ecart-type	*0:1:40.31*	-	*0:1:29.9*	-

Tableau E.4: Suite des données récupérées après les 6 manipulations du groupe 2.

E.2 Données des groupes 3 et 4

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
2005-06-17 11:28:16	0:5:57	5	0:1:18	21.85
2005-06-17 11:35:34	0:2:16	1	0:0:5	3.68
2005-06-17 11:49:54	0:2:21	1.67	0:0:16	11.35
2005-06-17 11:53:15	0:2:10	0.67	0:0:5	3.85
2005-06-17 11:56:07	0:3:9	1.33	0:0:8	4.23
Moyenne	*0:3:10.6*	*1.93*	*0:0:22.4*	*11.75*
Ecart-type	*0:1:25.8*	*1.57*	*0:0:28.09*	-

Tableau E.5: Données récupérées après les 5 manipulations du groupe 3.

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
2005-06-17 11:28:16	0:1:5	18.21	0:3:34	59.94
2005-06-17 11:35:34	0:0:24	17.65	0:1:47	78.68
2005-06-17 11:49:54	0:1:11	50.35	0:0:54	38.3
2005-06-17 11:53:15	0:0:43	33.08	0:1:22	63.08
2005-06-17 11:56:07	0:0:10	5.29	0:2:51	90.48
Moyenne	*0:3:10.6*	*22.35*	*0:2:5.6*	*65.9*
Ecart-type	*0:1:25.8*	*1.57*	*0:0:28.09*	-

Tableau E.6: Suite des données récupérées après les 5 manipulations du groupe 3.

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
2005-06-17 15:41:33	0:5:3	7.33	0:0:18	5.94
2005-06-17 15:50:20	0:3:46	3.33	0:0:8	3.54
2005-06-17 16:02:20	0:2:27	2	0:0:3	2.04
2005-06-17 16:05:38	0:2:36	0.67	0:0:8	5.13
Moyenne	*0:3:28*	*3.33*	*0:0:9.25*	*4.45*
Ecart-type	*0:1:2.8*	*2.49*	*0:0:5.45*	-

Tableau E.7: Données récupérées après les 4 manipulations du groupe 4.

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
2005-06-17 15:41:33	0:2:25	47.85	0:2:20	46.2
2005-06-17 15:50:20	0:0:37	16.37	0:3:1	80.09
2005-06-17 16:02:20	0:0:16	10.88	0:2:8	87.07
2005-06-17 16:05:38	0:0:13	8.33	0:2:15	86.54
Moyenne	*0:3:28*	*25.36*	*0:2:26*	*70.19*
Ecart-type	*0:0:54.06*	-	*0:0:20.65*	-

Tableau E.8: Suite des données récupérées après les 4 manipulations du groupe 4.

E.3 Données des groupes 5 et 6

Date	Durée	Mess./Pers.	Durée comm	
			Temps	%
2005-09-13 12:06:00	0:6:19	5.33	0:0:36	9.5
2005-09-13 12:15:00	0:3:2	1.33	0:0:6	3.3
2005-09-13 12:26:10	0:2:46	1	0:0:20	12.05
2005-09-13 12:30:23	0:2:6	1	0:0:11	8.73
Moyenne	*0:3:33.25*	*2.17*	*0:0:18.25*	*8.56*
Ecart-type	*0:1:37.85*	*1.83*	*0:0:11.41*	-

Tableau E.9: Données récupérées après les 4 manipulations du groupe 5.

Date	Durée coor		Durée prod	
	Temps	%	Temps	%
2005-09-13 12:06:00	0:1:50	29.02	0:3:53	61.48
2005-09-13 12:15:00	0:0:48	26.37	0:2:8	70.33
2005-09-13 12:26:10	0:0:31	18.67	0:1:55	69.28
2005-09-13 12:30:23	0:0:11	8.73	0:1:44	82.54
Moyenne	*0:0:50*	*23.45*	*0:2:25*	*68*
Ecart-type	*0:0:37.03*	-	*0:0:51.51*	-

Tableau E.10: Suite des données récupérées après les 4 manipulations du groupe 5.

Date	**Durée**	**Mess./Pers.**	**Durée comm**	
			Temps	%
2005-09-14 13:49:10	0:5:17	6.67	0:1:21	25.55
2005-09-14 13:55:29	0:4:31	4.67	0:0:15	5.54
2005-09-14 14:08:22	0:2:36	1.67	0:0:8	5.13
2005-09-14 14:13:17	0:6:42	6.33	0:0:14	3.48
Moyenne	*0:4:46.5*	*4.83*	*0:0:29.5*	*10.3*
Ecart-type	*0:1:28.8*	*1.98*	*0:0:29.85*	-

Tableau E.11: Données récupérées après les 4 manipulations du groupe 6.

Date	**Durée coor**		**Durée prod**	
	Temps	%	**Temps**	%
2005-09-14 13:49:10	0:1:4	20.19	0:2:52	54.26
2005-09-14 13:55:29	0:1:6	24.35	0:3:10	70.11
2005-09-14 14:08:22	0:0:41	26.28	0:1:47	68.59
2005-09-14 14:13:17	0:3:25	51	0:3:3	45.52
Moyenne	*0:1:34*	*32.81*	*0:2:43*	*56.89*
Ecart-type	*0:1:4.83*	-	*0:0:32.96*	-

Tableau E.12: Suite des données récupérées après les 4 manipulations du groupe 6.