Vers un Support des Communications Multi-Parties pour les Systèmes Multi-Agents

Julien Saunier, Flavien Balbo

Vers un Support des Communications Multi-Parties pour les Systèmes Multi-Agents

J. Saunier* F. Balbo*,†
saunier@lamsade.dauphine.fr balbo@lamsade.dauphine.fr

*LAMSADE, Université Paris-Dauphine
Place du Maréchal de Lattre de Tassigny
75775 Paris cedex 16 – FRANCE

†Institut National de Recherche sur les Transports et leur Sécurité
2 avenue du Général Malleret-Joinville
F-94114 ARCUEIL Cedex – FRANCE

Résumé :
Bien que les dialogues bi-parties soient les plus étudiés dans la communauté des Systèmes Multi-Agents (SMA), certains nouveaux modèles tels que l'écoute flottante et les communications multi-parties ont émergé récemment. Ces modèles ont montré des gains d'efficacité et de cohérence des SMA. Dans cet article, nous introduisons un cadre générique pour le support des communications multi-parties. Nous décrivons un modèle formel d'environnement appelé EASI (Environment as Active Support of Interaction). Des algorithmes pour la mise en oeuvre effective du modèle sont également proposés, et nous discutons la validité de cette approche à travers une série de tests.

Mots-clés : Interaction, environnement, communications multi-parties

Abstract:
Although two-party dialogues are the most-studied communication type in the Multi-Agent Systems (MAS) community, new models such as overhearing and multi-party communications have emerged recently. These models have been shown to improve the efficiency and the coherence of the MAS. In this article, we introduce a generic framework for multi-party communication support. Then we introduce a formal environment model called EASI (Environment as Active Support of Interaction). We also propose algorithms to support effectively this model, and we discuss the validity of this approach through a series of tests.

Keywords: Interaction, Environment, multi-party communications

1 Introduction

La forme de communication la plus étudiée est le dialogue, dans lequel deux participants échangent les rôles d'émetteur et de destinataire. Cependant, des travaux récents, inspirés par l'étude de situations réelles, mettent en exergue la possibilité d'exploiter des formes de communications plus complexes, impliquant de nouveaux rôles. Ainsi, l'écoute flottante [6] est l'écoute de communications entre d'autres agents, sans être impliqué directement dans la communication ni même nécessairement que les participants en ait connaissance.

En section 2, nous introduisons de façon générique les communications multi-parties. Nous décrivons en section 3 le modèle formel EASI (Environment as Active Support of Interaction). En section 4, nous introduisons des algorithmes de gestion de ce modèle qui dépendent de la dynamique du SMA et nous montrons une implémentation et discutons la performance du modèle et des algorithmes.

2 Les Communications Multi-Parties

De façon générale, les communications multi-parties présentent la spécificité d'avoir un émetteur, des destinataires prévus et des récepteurs inattendus. Une étude plus fine est nécessaire pour extraire les différents types de rôles possibles.

Sur la base des travaux existants dans

le domaine des SMA [5], mais aussi de la psychologie et des sciences sociales [2], nous proposons 3 critères déterminant le rôle de l'agent dans la communication : (1) l'*intention* : le récepteur est-il prévu, et si oui pour participer activement à l'échange ou pour l'écouter passivement ? (2) La *connaissance* : l'agent est-il connu des autres participants ? (3) L'*initiateur* de la réception : est-ce une initiative de l'émetteur ou du récepteur ? Ces critères vont nous permettre de définir les principaux rôles pouvant être joués dans la communication. Ainsi, le *destinataire* est un récepteur prévu et connu par l'émetteur, qui participe au dialogue, et pour lequel la réception s'est effectuée à l'initiative de l'émetteur. Un *auditeur* aura les mêmes caractéristiques, hormis qu'il n'est pas sensé participer à la conversation. Un *écouteur* est un récepteur non-prévu, qui peut ne pas être connu de l'émetteur, et qui perçoit le message par son initiative propre. Le cas d'un "groupe de destination" est le moins classique, il s'agit d'une initiative de l'émetteur envers un groupe dont les membres ne sont pas nécessairement connus. Par exemple, faire une annonce au club photographie de l'université n'implique pas de connaître chacun de ses membres. Enfin, un *écouteur indiscret* est un récepteur indésirable. Ce cas ne sera en général pas recherché, hormis par exemple dans le cadre de simulations.

Un système permettant la présence simultanée de destinataires et d'écouteurs doit résoudre la problématique de la mise en corrélation d'initiatives d'origines différentes pour la décision de distribution des messages. La possibilité de récepteurs imprévus, voir inconnus, implique que l'émetteur ne maîtrise pas nécessairement le canal de communication, par exemple une émission en clair sur un réseau wifi. Dans le cadre de réseaux classiques, ceci n'est réalisable que par le biais d'un middleware. Il est nécessaire de caractériser comment s'exprime l' "initiative" permettant la connexion entre l'information et les récepteurs. Une solution est de représenter chaque composant du SMA par une description observable, et de permettre aux agents d'utiliser ces descriptions pour gérer leurs interactions en ajoutant des conditions. Les composant du SMA sont les agents, les percepts (messages, traces) ou tout autre objet.

Exemple Illustratif. Au long de cet article, nous développons un exemple de service de communication dédié à une Application d'Intelligence Ambiante (AIA). Cette application doit faciliter les interactions entre les employés d'une entreprise et leurs visiteurs. Un agent *employé* appartient à un service, et a une disponibilité. L'entreprise est composée de salles de service, de salles de réunion et d'une réception. L'objectif sera de proposer une application permettant le support des différents besoins d'interaction de façon standardisée. Par exemple, pour un visiteur, les besoins d'interaction directe seront liés à la recherche d'un certain employé (situation notée S_{di1}), ou à la recherche d'un employé disponible dans un service déterminé (S_{di2}). Un exemple d'interaction indirecte sera lié à la libération d'une salle. Cet évènement devra être perçu par les agents intéressés (S_{ind}). Enfin, l'application doit supporter des modèles d'interaction plus complexes comme l'écoute flottante. Par exemple, un agent peut surveiller l'activité du SMA en écoutant les employés en présence de clients dans les salles d'un service particulier (S_{mon}).

3 L'Environnement, Support Actif de l'Interaction

L'originalité d'EASI est de modéliser l'interaction dans son ensemble et de considérer les agents comme une partie de cet ensemble. Le problème de connexion est résolu par l'environnement en fonction de la description qu'il a des composants de l'interaction, agents ou percepts ; et la réi-

fication de chaque problème de connexion sera appelée un *filtre*. Dans la suite, nous appellerons "connexion" la mise en relation d'un percept avec un ou plusieurs agent. Toutes les informations sur les composants du SMA nécessaires à l'interaction sont regroupées dans l'environnement. De façon à être facilement utilisable, ce regroupement doit reposer sur une organisation efficace, c'est pourquoi EASI est fondé sur l'Analyse de Données Symboliques (ADS) [1]. L'ADS est un modèle de classification et d'analyse de grands ensembles de données qualitatives, quantitatives et complexes.

Notre modèle d'environnement comprend un ensemble de m entités, $\Omega = \{\omega_1, ..., \omega_m\}$ et un ensemble de k filtres, $\mathcal{F} = \{f_1, ..., f_k\}$. Une entité ω_l possède une description d'un composant du SMA (agent, percept, objet) par le biais de ses propriétés observables. Un filtre f_j est la description de contraintes sur les propriétés observables des entités liées à un problème de connexion j, qui sera utilisé pour la transmission des percepts. Soit $P = \{p_1, ..., p_n\}$ l'ensemble des n propriétés observables d'un SMA, une propriété observable p_i est une fonction qui donne pour une entité ω_l une valeur : $\forall p_i \in \mathcal{P},\ p_i : \Omega \rightarrow d_i \cup \{unknown, null\}$, avec d_i le domaine de description de p_i. d_i peut être quantitatif, qualitatif, ou un ensemble fini de données.

La figure 1 décrit un exemple simple de notre modélisation pour l'AIA. Il y a quatre entités, $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4\}$ qui ont respectivement la description de l'agent *visiteur* v_1, des agents *employé* e_1 et e_2 et du message m_1. Les agents ont entre autres une propriété appelée *pos* (pour $position$), le domaine de description de cette propriété d_{pos} est l'ensemble des salles du bâtiment. La valeur de $pos(\omega_1)$ est la *réception* ; la valeur $pos(\omega_2)$ est $unknown$ car l'agent ω_2 ne l'a pas renseignée, enfin la valeur $pos(\omega_4)$ est $null$ car ω_4 ne possède pas cette propriété dans sa description. La valeur d'une propriété peut être modifiée dynamiquement lors de l'exécution, excepté pour $null$, qui exprime l'absence de cette propriété. Il y a trois filtres, $\mathcal{F} = \{di1, di2, ind\}$. Un agent devant résoudre un problème de connexion ajoute un filtre le décrivant dans l'environnement. Par exemple, les agents *employé* ont deux filtres en commun, $\{di1, di2\}$, et l'agent *employé* $e2$ a ajouté le filtre ind. De même que pour les filtres, les agents modifient l'environnement en ajoutant, retirant ou mettant à jour des entités. Par exemple, l'agent *visiteur* v_1 ajoute dans l'environnement l'entité ω_4, qui représente le message m_1. Grâce à l'ensemble des descriptions des composants du SMA, un processus d'appariement entre messages et agents (détaillé en section 4) permet de résoudre le problème de connexion pour m_1 via les filtres de $\mathcal{F}$.

Tous les composants du SMA sont au même niveau d'abstraction, c'est à dire des entités. Afin d'obtenir des catégories d'entités, nous utilisons l'information structurelle d'existence des propriétés observables (valeur de p_i égale à $null$). Ainsi, une catégorie est un sous-ensemble d'entités décrites par un même sous-ensemble de propriétés. Lors de la connexion, la classification est faite de manière précise en ajoutant des conditions que les enti-

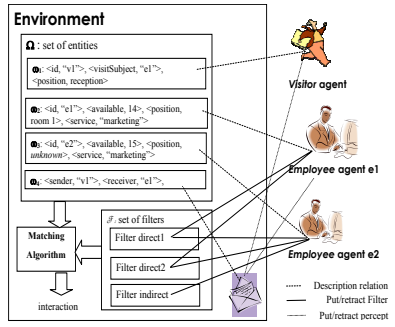

FIG. 1 – Exemple de modélisation des interactions avec EASI

tés doivent satisfaire. En ADS, un objet symbolique est une description cohérente d'une entité. Une assertion est un cas particulier d'objet symbolique, une conjonction de tests élémentaires. Une assertion est une description en intention, et son extension dans Ω contient toutes les entités satisfaisant cette description. Un filtre sera donc un objet symbolique décrivant les entités qui sont liées à un besoin en connexion particulier.

Definition 1 (Filtre) *Un filtre $f \in \mathcal{F}$ est un tuple $\langle f_{ag}, f_{pe}, [f_{co}], n_f \rangle$ où :*
- *f_{ag} est la description en intention de l'agent récepteur telle que : $a \in A, f_{ag}(a) = \wedge_{p_i \in P_{f_{ag}}} [p_i(a) R_{p_i}^{f_{ag}} d_{p_i}^{f_{ag}}]$.*
- *f_{pe} est la description en intention du percept telle que : $\omega \in \Omega$, $f_{pe}(\omega) = \wedge_{p_i \in P_{f_{pe}}} [p_i(\omega) R_{p_i}^{f_{pe}} d_{p_i}^{f_{pe}}]$.*
- *f_{co} est la description en intention (optionnelle) du contexte telle que : $C \subset \Omega, f_{co}(C) = \wedge_{\forall c \in C} f_{co}(c)$, avec $f_{co}(c) = \wedge_{p_i \in P_c} [p_i(c)\ R_{p_i}^{c}\ d_{p_i}^{c}]$.*
- *n_f est le nom du filtre.*

La description du récepteur (définition 1) est fondée sur les propriétés qu'un agent doit posséder pour être un récepteur potentiel. Avec $R_{p_i}^{f_{ag}}$, le tuple des opérateurs de comparaison et $d_{p_i}^{f_{ag}}$ le tuple des valeurs et variables, l'assertion f_{ag} décrit les conditions à satisfaire pour être récepteur. De la même façon, la description du percept à recevoir est donnée par l'assertion f_{pe}. Le contexte de l'interaction, i.e. les autres entités sur lesquelles portent des conditions, est donné par l'objet symbolique f_{co}. Le contexte est donc une partie de l'état observable du SMA. De façon à décrire une interaction, les deux premières assertions sont obligatoires.

Pour l'AIA, le filtre décrivant S_{di2} est : $f_{S_{di2}} = \langle [dep(a) = ?x] \wedge [ava(a) = true], [dep(io) = ?x], \emptyset, \text{“}di2\text{”} \rangle$
Les variables sont données préfixées avec un $?$, comme $?x$ dans l'exemple. Ce filtre décrit la condition sur les agents ($[dep(a) = ?x] \wedge [ava(a) = true]$), et sur les percepts ($[dep(io) = ?x]$). C'est un exemple de *groupe de destination* : l'émetteur choisit un groupe dont il connaît les critères, mais dont il ne connaît pas nécessairement les membres.

Le modèle EASI est générique car ce sont les filtres qui déterminent le modèle d'interaction utilisé, et tous les filtres sont traités de la même façon. Ainsi, les agents peuvent utiliser de façon standardisée n'importe quel modèle en fonction de leurs besoins. La distinction peut être effectuée en étudiant l'*initiateur* du filtre par rapport aux percepts. Soit f un filtre, si l'agent *initiateur* de f n'appartient pas à l'extension $E(f_{ag})$, ceci signifie qu'il n'est pas dans les récepteurs potentiels de ce filtre. C'est donc que le filtre décrit les agents avec lesquels l'*initiateur* désire interagir, ce qui est de l'interaction directe. Pour l'exemple AIA, les filtres de communication direct sont liés à S_{di1} et S_{di2}. La définition $f_{S_{di1}}$ est : $f_{S_{di1}} = \langle [id(a) = ?x], [receiver(io) = ?x], \text{“}di1\text{”} \rangle$
Dans ce cas, le récepteur est de type *destinataire*, car il est prévu et connu par l'émetteur, lequel a initié la connexion. Si l'*initiateur* fait partie de $E(f_{ag})$, c'est à dire qu'il fait partie des récepteurs potentiels de son filtre, alors f_{pe} décrit les percepts qu'il souhaite recevoir. C'est une interaction indirecte, à l'initiative du récepteur. L'exemple de filtre pour AIA d'interaction indirecte est : $f_{S_{ind}} = \langle [id(a) = \text{“}e2\text{”}], [pos(io) \in MR] \wedge [sub(io) = \text{“}available\text{”}], \text{“}ind\text{”} \rangle$
L'agent avec la propriété id à valeur “$e2$” percevra tous les percepts liés à la disponibilité des salles de réunion. C'est un *écouteur*, puisque c'est à son initiative qu'il accède aux informations. Enfin, pour les interactions de type écoute flottante, l'*initiateur* appartient à $E(f_{ag})$, mais le percept est initialement adressé à d'autres agents. L'exemple AIA sera : $f_{S_{mon}} = \langle [id(a) = \text{”}e3\text{”}], [sender(io) =$

$?y], [pos(ax) \in MR] \wedge [pos(ax) = ?x] \wedge [pos(ay) = ?x] \wedge [dep(ax) = \text{"}sav\text{"}] \wedge [id(ax) = ?y], \text{“}mon''\rangle$ avec $ax, ay \in A$. L'agent “$e3''$ est aussi un *écouteur* puisqu'il est à l'initiative du filtre et qu'il est récepteur potentiel.

4 Appariement

Une des difficultés du problème de connexion est de trouver un algorithme générique permettant de gérer les interactions quelle que soit la dynamique du SMA. Dans le cadre d'EASI, le critère principal d'évaluation de la dynamique est la fréquence de mise à jour des propriétés. Notre proposition est donc un algorithme d'appariement générique qui utilise les ensembles construits selon deux niveaux de description, l'existence des propriétés et les contraintes. L'algorithme d'appariement sera fondé sur la relation de validité suivante : $Soit\ a \in A, io \in IO, C \subset \Omega$, $V : A \times \Omega \times P(\Omega) \times \mathcal{F} \rightarrow \{true, false\}$, $V(a, io, C, f) = f_{ag}(a) \wedge f_{pe}(io) \wedge f_{co}(C)$

A chaque fois qu'une connexion est réalisée, un destinataire reçoit un percept. Autrement dit, lorsque $V(a, io, C, f)$ est valide, l'agent a reçoit Per_f, tel que $C' \subset C$, $Per_f = \{io, C', n_f\}$. L'ensemble des informations perçues en même temps que l'io est composé du nom du filtre, et d'un sous-ensemble du contexte de validation, i.e. une partie des entités de C. Un avantage d'EASI sera ainsi que le récepteur connaît le contexte dans lequel il reçoit un percept. Pour chaque percept ajouté dans l'environnement, l'algorithme doit associer les agents qui sont liés à ce percept par des filtres, en fonction du contexte. La réception effective est dénotée par la primitive $receive(a, Per_f)$, qui signifie la réception par l'agent a de l'ensemble de perception Per_f. Nous ne faisons aucune hypothèse sur l'architecture des agents. En considérant donc un ensemble PK_a, les connaissances privées de l'agent a, la primitive sera représentée algorithmiquement par : $receive(a, Per_f) \Leftrightarrow PK_a \leftarrow PK_a \cup Per_f$

Il faut trouver les ensembles les plus petits liés à chaque filtre. Une première solution est donc de ne calculer la validation que pour les entités possédant les propriétés requises. L'extension d'un filtre f est le tuple $\langle E(Pf_{ag}), E(P_{f_{pe}}), E(P_{f_{co}})\rangle$. Ces ensembles sont calculables pour une description du SMA donnée. Un percept peut être reçu par plusieurs agents grâce au même filtre. Par exemple, pour le même percept, $f_{S_{di2}}$ sera valide pour tous les agents disponibles du même service. De plus, un même percept peut être perçu grâce à plusieurs filtres. Par exemple, $f_{S_{di1}}$ et $f_{S_{mon}}$ peuvent être valides pour un même percept. La difficulté est donc de trouver pour un percept io tous les récepteurs potentiels, en fonction des filtres liés à cet io.

Nous définissons Cha_{io} comme l'ensemble des filtres f liés au percept io tel que cet io appartient à l'extension de chaque f (les définitions sont données en figure 2). Pour chaque filtre f dans Cha_{io}, on peut calculer l'ensemble des récepteurs potentiels et l'ensemble des contextes. Rec_{io} est l'ensemble des agents appartenant aux extensions des filtres appartenant à Cha_{io}, et Co_{io} est l'ensemble des contextes appartenant aux extensions de ces filtres. Enfin, sur le même principe, nous définissions $FPer_a$ (pour filtres de perception) comme l'ensemble des filtres

Nom	Définition
Cha_{io}	$\{f \in \mathcal{F} \mid io \in E(P_{f_{pe}})\}$
Rec_{io}	$\{a \in A \mid \exists f \in Cha_{io}, a \in E(P_{f_{ag}})\}$
Co_{io}	$\{C \subset \Omega \mid \exists f \in Cha_{io}, C \in E(P_{f_{co}})\}$
$FPer_a$	$\{f \in \mathcal{F} \mid a \in E(P_{f_{ag}})\}$
FCo_C	$\{f \in \mathcal{F} \mid C \in E(P_{f_{co}})\}$

FIG. 2 – Définitions des ensembles pour l'appariement structurel.

liés à un agent a, c'est à dire que l'agent appartient à l'extension de chacun de ces filtres. FCo_C (contextes de perception) est l'ensemble des contextes appartenant à l'extension de ces filtres. Chacun des ensembles est réduit aux entités et filtres potentiels. Par exemple, au lieu d'utiliser l'ensemble des agents A, nous avons Rec_{io}, i.e. le sous-ensemble des agents possédant les propriétés requises. Pour un percept io, un agent $a \in Rec_{io}$ et un contexte $C \in Co_{io}$, l'ensemble minimal des filtres pouvant effectuer la connexion est $(FPer_a \cap Cha_{io} \cap FCo_C)$. Cet algorithme limite la recherche d'appariement à l'espace des entités qui ont été classifiées en fonction de leur description en intention, ce qui améliore la résolution de la connexion. La valeur des propriétés n'étant pas prise en compte, ce niveau de description n'est pas sensible à la fréquence de mise à jour du SMA.

Algorithm 1 Algorithme d'appariement structurel

Pour chaque ($io \in IO$)
 Pour chaque ($a \in Rec_{io}$)
 Pour chaque ($C \in Co_{io}$)
 Pour chaque ($f \in (FPer_a \cap Cha_{io} \cap FCo_C)$)
 Si ($V(a, io, C, f)$) Alors
 $receive(a, Per_f)$
 Fin si
 Fin pour
 Fin pour
 Fin pour
Fin pour

Lorsque les propriétés observables ont un taux de mise à jour raisonnable, il est possible d'anticiper qu'un sous-ensemble de récepteurs potentiels, au sens possédant les propriétés requises, ne satisfont pas certaines conditions en terme de valeurs. Dans l'algorithme précédant, on évalue tout de même ces entités. Nous proposons donc un nouvel algorithme, qui tout en suivant le même déroulement sera fondé non plus sur les descriptions en intention, mais sur les extensions des descriptions des entités, i.e. $E(f_{ag})$, $E(f_{pe})$ et $E(f_{co})$ dans Ω. Ces extensions sont rarement calculables entièrement, car les appariements réalisés dans les filtres peuvent mettre en correspondance des propriétés de plusieurs ensembles d'entités. Nous proposons donc de modifier l'algorithme 1 en ôtant des ensembles d'appariement les entités dont la valeur des propriétés ne satisfait pas les conditions des filtres. Sur l'ensemble Cha_{io}, la sélection sera donc faite formellement par : $Cha^v_{io} = \{f \in Cha_{io} | \forall p_i \in P_{f_{pe}}[p_i(io) R_i^{f_{pe}} d_i^{f_{pe}}] \neq false\}$
Ceci signifie l'ensemble des filtres pour lesquels il n'y a pas de test élémentaire qui invalide l'io. Par exemple, $f_{S_{ind}}$ n'est valide que pour les io dont la propriété sub a pour valeur "$available$".

Par continuité, nous utiliserons dans ce second algorithme les ensembles restreints de récepteurs potentiels $Receiver^v_{io}$ et de filtres $FPer^v_a$ que l'agent peut satisfaire. Ce calcul peut être fait pour tous les ensembles de l'algorithme précédent. Il en résulte que le processus d'appariement est plus rapide grâce à un parcours d'ensembles plus petits, par contre le coût de maintenance des ensembles sera plus élevé. En effet, lorsqu'une entité met à jour ses propriétés, elle peut passer pour un filtre donné de "valide" à "invalide" ou inversement.

Expérimentations

De façon à évaluer la performance de nos algorithmes, nous avons mis en place une série de tests comparatifs comprenant la diffusion classique (Broadcast) et nos deux algorithmes, respectivement notés E1 et E2. Nous nous sommes intéressés en particulier à la dépendance entre le taux de mise à jour, le nombre d'agents et la performance du système. Les tests sont des simulations de l'exemple AIA décrit dans cet article.

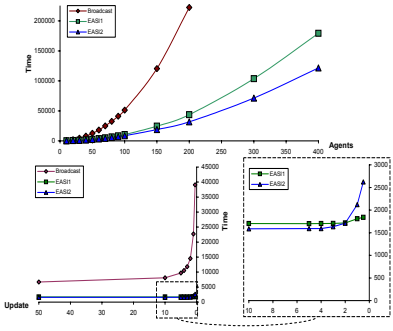

FIG. 3 – Broadcast - Algorithmes EASI. Temps d'exécution en fonction du nombre d'agent (haut) et du taux de mise à jour (bas).

Par broadcast, la résolution du problème de connexion est décentralisée, i.e. chaque agent calcule les récepteurs de ses messages, tandis qu'EASI centralise au niveau de l'environnement ce calcul. Pour pouvoir comparer ces deux approches, nous devons donc évaluer la performance du système dans son ensemble. Ainsi, à la fois le processus de décision des agents et la gestion de l'environnement sont mesurés dans un simulateur centralisé. Il est à noter que nous ne mesurons donc pas les coûts en bande passante des différentes solutions. A chaque pas de temps et dans un ordre aléatoire, chaque agent vérifie ses messages, puis choisit et exécute un comportement, comme répondre à un message, ajouter un filtre, etc. La moitié des agents sont des agents *employé*, l'autre moitié des agents *visiteur*. Chaque agent met à jour sa propriété ava lorsque c'est nécessaire, et les agents *employé* modifient leur propriété dep en fonction du taux de mise à jour.

Résultats. Les constantes de base sont un taux de mise à jour de 1/10 (une fois tous les 10 pas), pour 40 agents, sur 8000 pas. Le premier graphique (Fig. 3, haut) donne le temps d'exécution de la simulation en fonction du nombre d'agents. Le broadcast est le moins efficace quel que soit le nombre d'agents. C'est la limite classique du broadcast, qui le rend inutilisable si le nombre de messages et/ou d'agents devient important. Nous avons pu vérifier que nos alorithmes pouvaientt gérer un nombre assez important d'agents avec les deux algorithmes EASI : Nous avons exécutés des tests jusqu'à 1000 agents en 9 minutes, ce qui représente 22 millions de messages. Pour moins de 30 agents, le temps d'exécution d'E2 est plus long que celui d'E1, tandis que pour plus d'agents c'est le contraire. En effet, la création et la gestion des ensembles utilisés par E2 nécessitent plus de calculs que pour les ensembles utilisés par E1, tandis que l'appariement d'un message sera plus rapide pour E2. Pour un petit nombre d'agents, le surcoût du calcul des ensembles n'est pas rentabilisé par le gain de gestion, mais l'avantage d'E2 augmente à mesure du nombre d'agents.

Le second graphique (Fig. 3, bas et droite) montre le temps d'exécution en fonction du taux de mise à jour. A nouveau, le broadcast est clairement désavantagé par rapport à nos algorithmes. Puisqu'E1 utilise des ensembles calculés à partir de la classification structurelle des entités, le taux de mise à jour n'a pas d'effet sur l'algorithme lui-même. Finalement, le temps d'exécution d'E2 est sensible à une forte dynamique des propriétés : lorsque la fréquence de mise à jour est supérieure à 1/2 il devient moins efficace à cause du coût de mise à jour des ensembles. Ces tests montrent donc que le modèle EASI et nos algorithmes sont une solution valide au problème de connexion. Le choix entre E1 et E2 doit être fait en fonction de la taille et de la dynamicité du SMA.

5 Discussion

L'utilisation de l'environnement pour les interactions entre agents n'est pas nouveau, et d'autres travaux le modélisent comme espace commun et partagé dans

lequel les agents évoluent [7]. Nous partageons cette idée de placer l'environnement comme l'un des composant principaux de la conception du SMA. Cependant, ces travaux tendent à ne pas s'intéresser spécifiquement au problème des communications multi-parties. Notre modèle, quant à lui, unifie les différents moyens de communication de façon à les rendre utilisables conjointement, y compris dans le cadre d'agents purement interactionnels. Les modèles d'espaces de tuples, initiés par Linda [3], rentrent dans cette catégorie, même s'ils ne sont pas dédiés spécifiquement aux SMA. La différence avec notre modèle est que les agents (ou processus) ne sont pas représentés par des données au sein de l'environnement, et que l'utilisation d'objets symboliques pour retrouver l'information dans l'environnement permet une expressivité plus riche que les *templates* utilisés dans les modèles de type Linda. En effet, le système teste si une donnée considérée correspond à un template, tandis que les objets symboliques permettent les correspondances multiples entre entités distinctes. Notons que la programmation d'espaces de tuples peut être un moyen d'implémenter le modèle EASI.

Notre méthode de résolution des connexions permet le support effectif des communications multi-parties, grâce à la description de chaque entité du système par des propriétés observables. Notre proposition permet aux agents de déclarer explicitement leurs besoins dans l'environnement, lequel gère ensuite la connexion. De cette façon, les interactions directes, indirectes, aussi bien que les communications multi-parties peuvent être utilisées au sein d'un même SMA, de façon standardisée. Nous envisageons de tirer parti de l'expressivité du modèle pour l'étendre vers un système de normes et lois, de façon à contrôler les interactions. Par ailleurs, nous travaillons sur la complétion des tests réalisés pour prendre en compte les coûts en bande passante dans un contexte complètement distribué. Enfin, nous étudions l'application de notre modèle à la gestion de crise et plus généralement à la simulation.

Références

[1] H. Bock and E. Diday. Analysis of symbolic data. exploratory methods for extracting statistical information from complex data. In *Studies in Classification, Data Analysis, and Knowledge Organisation*, volume 15. Springer-Verlag, 2000.

[2] H. Branigan. Perspectives on multi-party dialogue. *Research on Language & Computation*, 4 (2-3) :153–177, October 2006.

[3] N. Carriero, D. Gelernter, and J. Leichter. Distributed data structures in linda. In *popl'86 : Proceedings of the 13th ACM Sigact-Sigplan symposium on Principles Of Programming Languages*, pages 236–242, 1986.

[4] J. Dugdale, J. Pavard, and B. Soubie. A pragmatic development of a computer simulation of an emergency call center. In *Designing Cooperative Systems : The Use of Theories and Models*, pages 241–256. IOS Press, 2000.

[5] S. Kumar, M. J. Huber, D. McGee, P. R. Cohen, and H. J. Levesque. Semantics of agent communication languages for group interaction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 42–47. AAAI Press / The MIT Press, 2000.

[6] E. Platon, N. Sabouret, and S. Honiden. Overhearing and direct interactions : Point of view of an active environment, a preliminary study. In *Proceedings of Environment for Multi-Agent Systems, Workshop held at the Fourth Joint Conference in Autonomous Agents and Multi-Agent Systems*, pages 121–138. Springer Verlag, 2005.

[7] D. Weyns, H. V. D. Parunak, F. Michel, T. Holvoet, and J. Ferber. Environments for multiagent systems, state-of-the-art and research challenges. *Lecture Notes in Computer Science Series*, 3374 :2–52, 2005.