une méthodologie pour la conception de simulateur multi-agents basée sur l'organisation

Fabien Michel, Olivier Gutknecht et Jacques Ferber

Laboratoire d'Informatique, Robotique et Micro-électronique de Montpellier.
C.N.R.S. – Université Montpellier II, 161 rue Ada 34392 Montpellier Cedex 5 - France
{fmichel, gutkneco, ferber}@lirmm.fr
http://www.lirmm.fr/~{fmichel, gutkneco, ferber}

Résumé : Cet article présente une méthodologie originale pour la conception de simulateur multi-agents basée sur le modèle organisationnel AALAADIN. Cette approche vise notamment à faciliter l'étude des problèmes liés à la gestion de l'exécution des agents (le problème du scheduling). L'idée principale est d'envisager le simulateur comme un système multi-agents (SMA) de façon à exploiter sa structure organisationnelle sous-jacente pour expliciter ce problème. Dans l'optique de cette méthodologie, nous proposons des outils de simulation qui sont génériques dans le sens où ils n'imposent aucune contrainte sur la politique d'exécution envisagée. Au contraire, leur but est de permettre l'élaboration de structures d'activation complexes qui restent compréhensibles, modifiables et donc analysables grâce à une décomposition logique du problème. Plus précisément, nous décrivons ici le fonctionnement d'un agent de la plate-forme MADKIT nommé *Scheduler* qui permet l'élaboration de telles politiques d'exécution.

1. Introduction

Comme Axtell l'a encore montré récemment (Axtell, 2000), les résultats d'une simulation multi-agents sont fortement influencés par la façon dont celle-ci est implémentée. Notamment, il est clair que la technique employée pour gérer l'exécution des agents (scheduling) a un impact crucial sur les résultats (Lawson & Park, 2000) (Huberman & Glance, 1993). Cette observation trouve une explication dans le fait que nous ne disposons actuellement pas de formalisme adéquat pour représenter simplement des interactions collectives. Plus particulièrement, nous éprouvons une difficulté à modéliser la simultanéité des actions des agents (Ferber, 1995).

Cependant, peu de travaux portent sur l'analyse des biais engendrés directement par ce problème. Cette problématique, qu'on désignera ici par *problème du scheduling* ou autrement dit la *politique d'exécution des agents*, n'a en effet quasiment aucun support méthodologique alors que, paradoxalement, c'est un passage obligé lors de l'implémentation d'un simulateur.

En effet, le paradigme multi-agents est fondé implicitement sur la composition de comportements individuels concurrents (Resnick, 1995). Par ailleurs, les architectures multiprocesseurs n'apportent pas de solution évidente : sans synchronisation les agents évoluent au rythme de la complexité de leur architecture interne, ce qui ne permet pas de contrôler la cohérence de la dynamique globale du système. Ainsi toutes les solutions envisagées pour simuler un système multi-agents (SMA) sont génératrices de biais dans le processus de simulation. Autrement dit l'implémentation de ce point précis a un impact direct sur l'évolution du modèle multi-agents considéré alors qu'on souhaiterait sa neutralité.

Sans apporter ici de solution à la modélisation de la simultanéité, nous pensons que la conception et l'analyse du scheduling doit prendre une place prépondérante lors de la conception d'un simulateur multi-agents. Il est donc impératif de se donner des moyens

méthodologiques et pratiques qui permettent une analyse simple et explicite de ce problème. Il ne s'agit pas de vérifier une hypothétique correspondance avec la réalité mais bien de mieux connaître l'outil d'expérimentation et ses implications dans le résultat final, notamment afin de pouvoir évaluer les approximations et les erreurs liées directement à celui-ci.

Dans la première partie de cet article nous précisons les enjeux de cette problématique et nous ferons une synthèse des techniques de scheduling actuelles ainsi que des problèmes qu'elles soulèvent. Dans la section 3, nous décrivons une approche organisationnelle du problème et nous présentons les outils de simulation génériques associés à la méthodologie proposée. La section 4 constitue quant à elle la conclusion.

2. Problématique

2.1 Le scheduling : au cœur du processus de simulation

Si de façon classique une simulation consiste à expérimenter des modèles donnés sous la forme de relations mathématiques entre des variables, au contraire la simulation multi-agents se propose de représenter directement les individus, leurs comportements et leurs interactions (Parunak et al., 1998). L'étude de la problématique liée à l'implémentation de ces systèmes complexes est en général centrée autour d'une ou plusieurs des notions suivantes : le type des agents simulés (communicant, situé, réactif, cognitif, …), l'environnement dans lequel évoluent les agents (discret, continu, 2D, 3D, …) et la nature des interactions (coordination, négociation, perceptions et actions sur l'environnement, …)

Une telle approche a l'inconvénient de ne pas faire apparaître de façon explicite les problèmes liés à l'implémentation de la simulation du temps. Alors qu'il est clair que chaque modèle ne nécessitera pas le même niveau de granularité temporelle : du temps réel pour la robotique mobile à la simulation de plusieurs années en écologie. Par ailleurs, simuler un SMA suppose qu'on a implémenté un mécanisme permettant de synchroniser les actions des agents. Σ définissant l'ensemble des états possibles du système, toute simulation est basée sur l'hypothèse que l'évolution du monde $\sigma \in \Sigma$ de l'instant t à $t+dt$ résulte de la composition des actions $A1(t)$, $A2(t)$, …, $An(t)$ produites par les agents à l'instant t. Autrement dit, il s'agit de construire une fonction du temps, *Dynamique D*, telle que :

$$D : \Sigma \rightarrow \Sigma$$
$$\sigma(t+dt) = D\left(\prod Ai(t), \sigma(t) \right)$$

Le symbole Π est ici utilisé pour désigner l'opérateur de composition des actions. Il définit de quelle manière les actions produites à un instant t doivent être sommées afin de calculer leurs conséquences sur l'état du monde. Sans entrer dans le détail, il est facile de mesurer la difficulté de conceptualiser une telle opération étant donné la multitude et la nature des concepts qui peuvent se cacher derrière le mot action (mouvement, prise de décision, modification de l'environnement). C'est pourquoi il n'existe pas de consensus sur la manière dont le déroulement des interactions doit être simulé et tous les concepteurs de simulation multi-agents sont amenés à faire un choix personnel en la matière. Et ce choix est difficile : un modèle multi-agents ne définit pas la façon dont il doit être implémenté, précisément parce qu'il s'agit d'un problème lié à l'outil informatique et non au modèle.

Ainsi un seul modèle théorique peut donner des résultats très différents suivant la structure d'activation utilisée. C'est pourquoi au contraire des mathématiques (dont la validité et la

constance ne dépendent pas d'un modèle) dans les simulations numériques, la technique de scheduling utilisée est indissociable des résultats obtenus.

De façon classique l'expérimentation sert à évaluer la qualité d'un modèle. Elle doit permettre de le valider ou de le remettre en cause en vue d'une modification. Nous pensons que dans le cadre de la simulation multi-agents, il est impératif d'inclure la politique d'exécution dans ce processus d'évaluation.

2.2 Les techniques de scheduling actuelles

Nous présentons ici de façon non exhaustive trois méthodes de synchronisation utilisées dans les plates-formes actuelles : à *pas de temps constant, double buffer* et *par événements*.

2.2.1 Les simulations à *pas de temps constant*

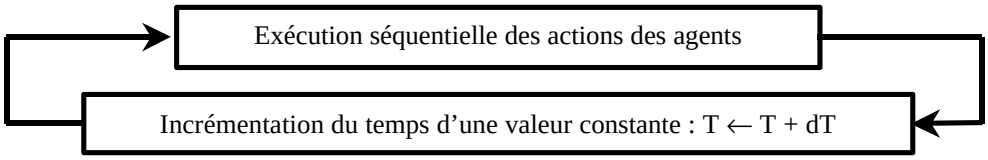

Figure 1. Principe d'une simulation à pas de temps constant

Comme le montre la figure 1, cette méthode consiste à activer les agents séquentiellement. L'activation de l'ensemble du système correspond alors à un pas de temps. Cette technique a l'avantage d'éviter la présence de conflits dans l'accès aux variables de l'environnement et de garantir que tous les agents agissent une fois par cycle. Par ailleurs, c'est de loin la technique la plus utilisée du fait de sa simplicité. Par exemple, les plates-formes STARLOGO (Resnick, 1995) et MANTA (Drogoul & Ferber, 1992) utilisent cette technique.

Notre objet n'est pas ici d'énumérer tous les biais engendrés par ce type de fonctionnement. Nous allons cependant illustrer notre propos à l'aide de deux exemples simples. Le premier décrit un modèle de simulation connu sous le nom *proies/prédateurs*. On pose qu'une proie (un triangle) est capturée lorsqu'elle est entourée de quatre prédateurs (les ronds). On voit sur la figure 2 que l'ordre d'exécution des agents (ici représenté par un numéro) peut changer l'issue d'une même situation. Ici la vie ou la mort de la proie.

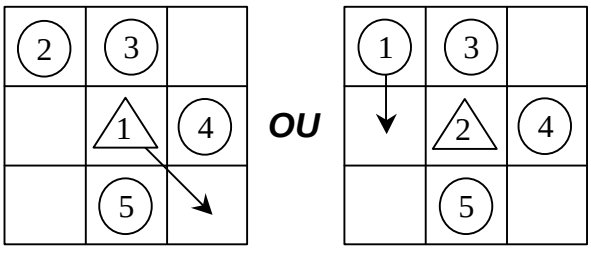

Figure 2. La survie de la proie dépend de sa place dans la liste

Notre deuxième exemple met l'accent sur le problème de la granularité des actions. Cet exemple est tiré du modèle de simulation de la plate-forme STARLOGO. Une tortue (la dénomination des agents dans ce modèle) peut effectuer à chaque pas de temps une action primitive. Ici la primitive considérée est *fd n*. Elle signifie que la tortue avance de *n* cases.

La figure 2 montre que, *n* n'étant pas fixé, plusieurs tortues peuvent avoir des trajectoires qui se croisent sans être à aucun instant sur la même case[1], la vitesse des déplacements n'ayant pas de lien direct avec le temps.

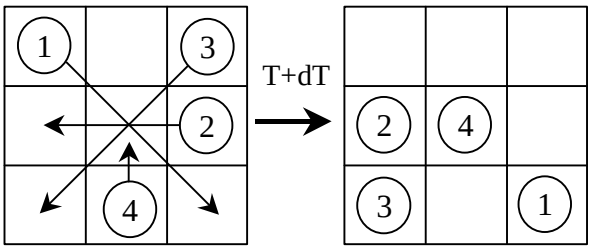

Figure 3. Le problème de la granularité temporelle des actions

2.2.2 Simulation avec état tampon ou « double buffer »

Extension de la précédente, cette technique vise à apporter une solution au problème de la simultanéité. L'objectif est de faire en sorte qu'à un instant *t* tous les agents aient la même perception de l'état du monde σ(t). Pour cela les actions des agents se font sur des variables tampons et le monde n'est pas directement modifié. Une fois tous les agents exécutés, on fait la synthèse de l'ensemble des actions pour calculer le nouvel état du monde.

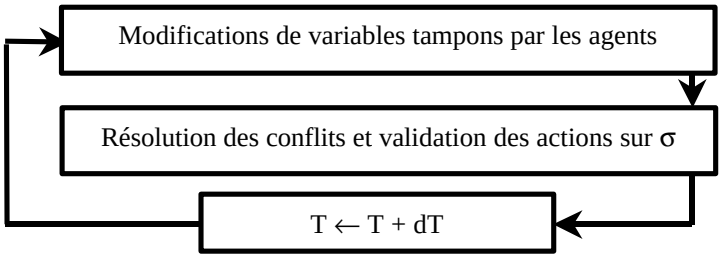

Figure 4. Un modèle de simulation avec état tampon

Les simulations de type *jeu de la vie* sont basées sur ce principe : l'état du monde est mis à jour une fois que toutes les cellules ont calculé leur état suivant. La plate-forme de Travers, LIVEWORLD (Travers, 1996), propose ce mode de fonctionnement. Cette méthode entraîne cependant directement un nouveau problème. Un conflit apparaît lorsque deux agents ou plus spécifient de nouvelles valeurs différentes pour une même variable. Il faut donc éventuellement résoudre les conflits aboutissant à une incohérence de l'état du monde.

Par ailleurs, Travers, y voit lui-même un autre sérieux désavantage: si un agent modifie une variable, il ne peut pas s'en servir pour un nouveau calcul sans risquer d'obtenir un résultat incohérent, la valeur de la variable n'étant validée qu'à la fin d'un cycle. Sans entrer dans le détail des inconvénients liés aux résolutions de conflits, on peut simplement remarquer qu'on en revient à instaurer un mécanisme de priorité entre les agents lors de la résolution d'un conflit. Ainsi, même si ce procédé se fonde sur une analyse plutôt que sur une méthode empirique, on peut facilement retrouver des situations semblables aux précédents exemples.

[1] On voit aussi que si la tortue 4 s'était déplacée en première, les autres l'auraient trouvée sur leur trajectoire.

2.2.3 Simulation par événements

Plutôt que de synchroniser tous les agents à un instant *t*, l'idée de cette méthode est d'exhiber explicitement un ordre chronologique entre les actions des agents. Le principe consiste à déterminer a priori ou en cours de simulation les événements futurs, leur date et leur nature.

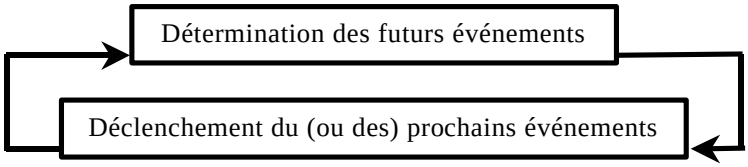

Figure 5. Principe d'une simulation par événements

Magnin utilise par exemple dans son simulateur SIEME (Magnin, 1996) un ensemble de règles environnementales permettant de déterminer au fur et à mesure des interactions les événements à déclencher: *Si <condition événement> alors <déclencher événement>*. Ainsi un principe de causalité entre événements est au cœur de ses préoccupations. Au contraire dans la plate-forme SWARM (Minar et al., 1999) la chronologie des interactions est déterminée a priori sous la forme d'une collection structurée d'objets à laquelle est associé un ordonnancement. Celui-ci définit alors la logique de la dynamique du système.

Bien que ce type de simulation semble loin des précédents, il n'en reste pas moins qu'on retrouve les mêmes genres de problèmes. En effet, lorsque plusieurs événements sont concurrents les difficultés liées à la modélisation de la simultanéité restent les mêmes.

2.3 Discussion

Nous pensons les difficultés liées à la conception d'une simulation ne tiennent pas particulièrement aux biais engendrés par une technique précise de scheduling. Comme nous l'avons dit, il faut faire un choix. Nous pensons plutôt que le simple fait de faire en la matière un choix unilatéral est une erreur. Il faut remarquer que le fonctionnement d'une simulation est toujours basé sur une unique politique d'exécution. Ainsi tous les agents de la simulation sont en général soumis à un seul et même principe de simulation. A partir de là, la tâche peut s'avérer excessivement complexe dès lors qu'on s'attaque à un modèle comportant des agents très différents. En effet la granularité des actions (mouvement, vitesse, etc.) et la sémantique des interactions (collisions, coordination, etc.) peuvent être très différentes.

De plus, il n'est pas rare que les processus qui accompagnent la simulation (affichages, analyse statistique, etc.) soient eux aussi gérés sur le même principe. Dans un système événementiel, l'affichage correspond à un événement qu'il faut gérer de la même façon que les autres. Ainsi le fait que tout le système soit soumis à la même politique contribue à rendre son analyse difficile et limite ses possibilités d'extension à d'autres agents.

3. Le simulateur comme un SMA

3.1 Organiser pour régner

Notre idée est simple et d'ailleurs elle n'est pas nouvelle dans le principe. Il s'agit de diviser le problème du scheduling global en autant de sous problèmes que nécessaire. En effet il suffit de remarquer que – quelle que soit la méthode – synchroniser tous les agents de la même

manière n'a plus vraiment de sens lorsqu'on est amené à considérer plusieurs types d'agents, chacun de ces types pouvant nécessiter l'emploi d'une technique de scheduling différente.

Il est plus simple de se concentrer sur un groupe d'agents dont la synchronisation est cruciale pour le processus de simulation. Ainsi, en élaborant des groupes formés d'agents dont la synchronisation des actions est nécessaire, on décompose le problème global : chaque groupe peut être traité indépendamment de façon à identifier un protocole de scheduling adapté à la situation. De plus, ces groupes définissent une organisation multi-agents dont la nature reflète de façon explicite la logique du simulateur.

3.2 Le modèle Aalaadin et la plate-forme MADKIT

Nous présentons ici le modèle organisationnel AALAADIN (Ferber & Gutknecht, 1998) et la plate-forme MADKIT (Gutknecht et al., 2000) basée sur celui-ci. Ce modèle est basé sur trois concepts clés : *agent*, *groupe* et *rôle* (AGR).

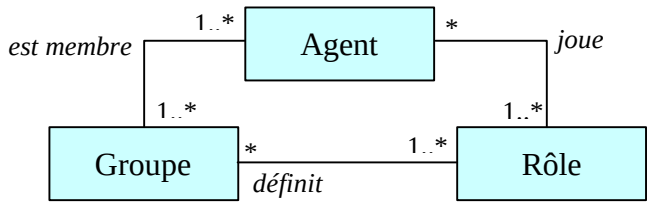

Figure 6. Le modèle Agent/Groupe/Rôle

Agent : Quasiment aucune contrainte n'est posée sur l'architecture interne ou sur le modèle de l'agent. L'*agent* est simplement décrit comme une entité autonome communicante qui joue des *rôles* au sein de différents *groupes*. Cette très faible sémantique est volontaire. Le but est de laisser toute liberté au concepteur pour choisir l'architecture appropriée à ses besoins.

Groupe : Le groupe est la notion primitive de regroupement d'agents. Chaque agent peut être membre d'un ou plusieurs groupes. D'une façon un peu simpliste, un groupe peut être vu comme un moyen d'identifier par regroupement un ensemble d'agents. Plus classiquement, associé au rôle, il définira la structuration organisationnelle d'un SMA usuel. Les différents groupes peuvent par ailleurs se recouper librement.

Rôle : Le rôle est la représentation abstraite d'une fonction, d'un service ou d'une identification d'un agent au sein d'un groupe particulier. Chaque agent peut avoir plusieurs rôles, un même rôle peut être tenu par plusieurs agents, et les rôles sont locaux aux groupes.

La structure organisationnelle que nous venons de décrire est implémentée au cœur de la plate-forme MADKIT, tant pour fournir un modèle organisationnel aux SMA exécutés que pour le fonctionnement interne du système. De plus MADKIT intègre deux principes de conception supplémentaires : une architecture à micro-noyau et l'agentification systématique des services.

3.3 L'agent Scheduler de MADKIT

Notre idée est d'avoir transposé à la simulation les principes de conception de la plate-forme MADKIT. Nous avons donc implémenté un agent intégré au micro-noyau qui permet donc la conception de techniques hétérogènes de scheduling : l'agent *Scheduler*. Son rôle consiste à manipuler **des** politiques d'exécution sur lesquelles aucune contrainte n'est posée.

Pour cela, cet agent est associé à un objet outil générique appelé *Activateur*. Dans sa forme la plus simple un activateur est simplement le moyen pour le Scheduler d'identifier un ensemble d'agents étant donné un groupe et un rôle. Par exemple, le Scheduler peut créer un activateur sur le rôle *agent* dans le groupe *simulation*, ou encore *afficheur* dans le groupe *représentation graphique*. De plus un même agent Scheduler peut créer autant d'activateurs que nécessaire. Sa tâche se résume à ordonner l'exécution de ses activateurs pour définir le processus de simulation dans son ensemble. Ainsi, le principe est de spécialiser, si nécessaire, de nouvelles sous-classes d'Activateur de façon à définir une technique de scheduling particulière qui pourra ensuite être appliquée ponctuellement, par le Scheduler, à différents groupes d'agents.

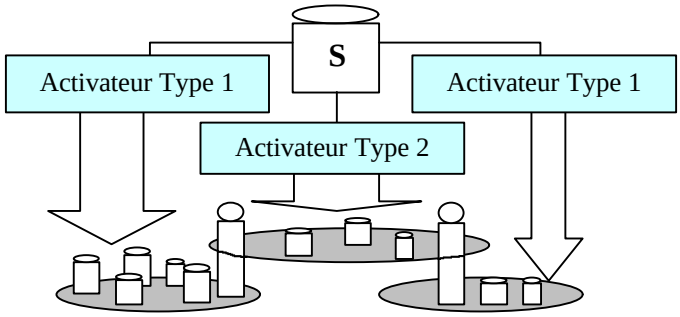

Figure 7. Un simulateur « organisationnel »

La figure 7 décrit par exemple un simulateur comportant un agent Scheduler *S* qui utilise deux types d'activateur (deux politiques d'exécution différentes) sur trois groupes d'agents. De plus cette figure montre qu'un agent peut appartenir à plusieurs groupes simultanément.

Le principal avantage de la décomposition du scheduling global réside dans la possibilité de modifier ou de remplacer un activateur sans avoir à toucher la structure globale : on a découplé les problèmes de gestion de la simulation (Scheduler) de celui de la synchronisation des agents (Activateur), le Scheduler n'étant pas responsable de la qualité d'un activateur. Un agent Scheduler peut de plus, en tant que simple agent, faire l'objet d'un contrôle supérieur.

Par ailleurs ce mécanisme ne fait aucune supposition sur l'architecture interne des agents. Ainsi, un agent (une classe JAVA par exemple) peut être remplacé par un autre de façon transparente, sans avoir à modifier le Scheduler ou ses activateurs. De la même façon, un agent peut être lancé en cours d'exécution : le simple fait de jouer un rôle précis engendre la participation de celui-ci au processus de simulation. Il est par exemple possible de compiler un agent durant la simulation pour ensuite l'incorporer à celle-ci sans la stopper.

3.4 Méthodologie de conception d'un simulateur organisationnel

Nous proposons une méthodologie de conception de simulateur multi-agents en trois étapes :

1. Exprimer la structure organisationnelle du simulateur sous forme de groupes et de rôles.
2. Elaborer les types d'activateurs nécessaires au simulateur (c'est-à-dire des techniques de scheduling locales).
3. Définir le fonctionnement global du simulateur en ordonnant l'exécution des activateurs.

L'énoncé de la troisième étape peut laisser croire que nous nous trouvons face à la même problématique que celle liée à l'ordonnancement des agents. Comme nous l'avons dit, il s'agit bien de deux niveaux d'analyse distincts. En effet un activateur, lié à un groupe, définit une technique de scheduling particulière (à pas de temps constant, par événements, …) pour des agents dont la synchronisation est considérée comme nécessaire. Au contraire, l'ordre d'exécution des activateurs décrit simplement la logique du simulateur, c'est-à-dire l'ordre dans lequel les différents groupes clés (agents simulés, affichage, observation, etc.) interviennent dans le processus de simulation.

La figure 8 illustre cet aspect de notre approche. Elle décrit un exemple d'ordonnancement de quatre activateurs différents A1, A2, A3 et A4. Chaque activateur est lié à un ensemble d'agents identifié par un groupe et un rôle. Les politiques de scheduling utilisées par les différents activateurs sont indépendantes. A1 peut par exemple utiliser une méthode à pas de temps constant alors que A2 définit un principe événementiel.

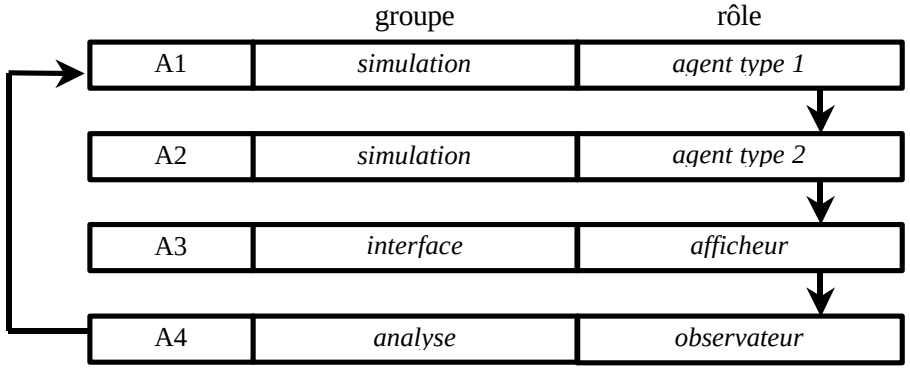

Figure 8. Exemple d'application du principe de simulation par activateurs

Cependant il est évident que l'ordre d'exécution des activateurs peut influer sur les résultats. Ainsi si on inverse A1 et A2, on peut obtenir des résultats différents étant donné que ces activateurs manipulent des agents (agent type1 et agent type2) qui peuvent modifier le monde. Néanmoins il ne s'agit pas ici d'un problème de synchronisation : la décomposition en deux activateurs implique qu'on suppose que les différents groupes d'agents ne doivent pas intervenir en même temps.

Par contre, il est tout à fait envisageable de considérer, après analyse des résultats, que la synchronisation de deux groupes soit finalement nécessaire. Il est alors très intéressant de pouvoir changer localement le fonctionnement du simulateur. On peut par exemple imaginer la fusion de A1 et A2 en un seul activateur plus adéquat, A3 et A4 restants valides. L'interchangeabilité des activateurs, obtenue grâce à la vue organisationnelle, trouve alors tout son sens : elle permet de constituer rapidement plusieurs simulations basées sur différentes techniques de scheduling sans avoir à recoder les agents de la simulation.

Nous soutenons qu'une telle méthodologie permet de construire une politique d'exécution très complexe au niveau global qui reste compréhensible, et donc analysable, grâce à la décomposition du problème. De plus, nous pensons que la possibilité de pouvoir modifier **localement** le fonctionnement du simulateur permet d'envisager la réalisation de simulations toujours plus complexes comportant un grand nombre d'agents fortement hétérogènes.

3.5 Applications

L'ensemble des simulations développées par notre équipe SMA se fonde sur ces outils. Des plus simples aux plus complexes comme le TURTLEKIT de MADKIT. Ce dernier est une librairie de simulation pour agents réactifs qui copie une partie des fonctionnalités de la plate-forme STARLOGO. Bien que le TURTLEKIT semble être une simple copie de STARLOGO, le fonctionnement du simulateur est en fait entièrement géré par des agents. Ainsi, sachant que MADKIT n'impose aucune contrainte sur l'architecture utilisée, les agents de la simulation peuvent librement être redéfinis par l'utilisateur (interface graphique, outil d'analyse, etc.).

Par ailleurs, l'utilisation du modèle AGR pour gérer les communications dans la plate-forme MADKIT, donne la capacité aux agents d'un simulateur de pouvoir communiquer avec tous les agents actifs, et pas seulement avec les agents de leur propre simulation. Ainsi il est possible d'utiliser toutes les fonctionnalités et outils offerts par la plate-forme MADKIT comme l'agent *MessageTracer* qui permet d'espionner les échanges de messages entre agents.

Loin de cet exemple, Simonin utilise actuellement le moteur synchrone de MADKIT pour simuler des robots mobiles autonomes (Simonin & Ferber, 2000). Cela confirme que ces outils de simulation ne posent pas de contrainte sur le modèle envisagé mais simplement des outils de conception qui peuvent être réutilisés sans peine.

4. Conclusions

Dans cet article nous avons vu que le scheduling est un point clé dans la simulation d'un SMA. Ainsi, notre but principal étant de proposer des outils génériques pour la conception de simulateur, ceux que nous avons présentés ici ne sont pas liés à une technique de scheduling particulière. De plus ces outils faisant parti du micro-noyau de MadKit, ils n'imposent aucune contrainte sur l'architecture des agents.

Par ailleurs, nous avons défendu l'idée que l'utilisation explicite de la structure organisationnelle d'un SMA est une facilité pour la conception de simulateur multi-agents. La méthodologie que nous avons présentée ici est basée sur cette idée. Nous soutenons qu'une telle méthodologie permet d'envisager l'édification de politiques complexes qui restent compréhensibles, et donc analysables, grâce à une localisation des problèmes de synchronisation.

Notre objectif à long terme est de proposer de nouvelles techniques de simulation qui permettront, par leur simplicité et leur interchangeabilité, d'expérimenter plus avant un modèle multi-agents dans le cadre de la simulation. Il semble par exemple tout à fait pertinent d'imaginer expérimenter un même modèle suivant différentes politiques de manière à évaluer leur impact sur les résultats obtenus. Nous pensons que cela permettra de faire un pas de plus vers une analyse plus approfondie et plus fructueuse des simulations multi-agents.

Références

AXTELL R. L. (2000). *Effects of Interaction Topology and Activation Regime in Several Multi-Agent Systems*. In the Second Workshop on Multi Agent Based Simulation. MABS-2000 (LNAI 1979), Juillet 2000.

DROGOUL A. and FERBER J. (1992). *Multi-Agent Simulation as a Tool for Modeling Societies: Application to Social Differentiation in Ant Colonies*. In Proceedings of MAAMAW'92, Viterbo, 1992.

FERBER J. (1995). *Les systèmes multi-agents. Vers une intelligence collective*. Inter-éditions 1995.

FERBER J. & GUTKNECHT O. (1998). *A meta-model for the analysis and design of organizations in multi-agent systems*. In Proceedings of Third International Conference on Multi-Agent Systems (ICMAS' 98), pp. 128-135, IEEE Computer Society, 1998.

GUTKNECHT O., FERBER J. et MICHEL F. (2000). *MadKit: Une expérience d'architecture de plate-forme multi-agents* générique. Huitièmes Journées Francophones Intelligence Artificielle Distribuée Systèmes Multi-Agents JFIADSMA 2000. Saint-Jean Le Vêtre Octobre 2000.

HUBERMAN B. A. & GLANCE. N. S. (1993). *Evolutionary Games and Computer Simulations*. Proceedings of the National Academy of Science USA, 90(August):7716-7718, 1993.

LAWSON B. G. & PARK S. (2000): *Asynchronous Time Evolution in an Artificial Society Mode*. In Journal of Artificial Societies and Social Simulation vol. 3, no. 1. http://www.soc.surrey.ac.uk/JASSS/3/1/2.html

MAGNIN L. (1996). *Modélisation et simulation de l'environnement dans les systèmes multi-agents: application aux robots footballeurs*. Thèse de doctorat, université Paris VI, 1996.

MINAR N., BURKHART R., LANGTON C. et ASKENAZI M. (1996). *The Swarm Simulation System, A Toolkit for Building Multi-Agent Simulations*. http://www.swarm.org/intro-papers.html

PARUNAK H. V. D., SAVIT R. et RIOLO R. L. (1998). *Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide*. In Proceedings of Workshop on Modelling Agent Based Systems. MABS' 98, Paris, 1998.

RESNICK M. (1995). *New Paradigms for Computing, new Paradigms for Thinking*. Computers and Exploratory Learning, edited by A. diSessa, C. Hoyles, and R. Noss. Springer-Verlag, 1995.

SIMONIN O. & FERBER J. (2000). *Modeling Self Satisfaction and Altruism to handle Action Selection and Reactive Cooperation*. In the proceedings of The Sixth International Conference on Simulation Adaptative Behavior SAB-2000, pp. 314-323, Paris, Septembre 2000.

TRAVERS M. D. (1996) *Programming with Agents: New metaphors for thinking about computation*. Thesis, Massachusetts Institute of Technology, pp. 127-137, 1996.