De la simulation multi-agents à la simulation multi-niveaux. Pour une réification des interactions.

Sébastien Picault

De la simulation multi-agents à la simulation multi-niveaux

Pour une réification des interactions

Mémoire d'Habilitation à diriger des recherches
de l'Université Lille 1
Spécialité : Informatique
présenté par :

Sébastien Picault

Laboratoire d'Informatique Fondamentale de Lille

Présentation prévue le 6 décembre 2013
Composition du jury :

Yves Demazeau	DR CNRS, Grenoble	Rapporteur
Amal El Fallah Seghrouchni	Pr., Université Pierre et Marie Curie	Rapporteur
Jacques Ferber	Pr., Université Montpellier II	Rapporteur
Rémy Courdier	Pr., Université de La Réunion	Examinateur
Anne Ruas	Chercheur, IFFSTAR	Examinateur
Sophie Tison	Pr., Université Lille 1	Présidente
Philippe Mathieu	Pr., Université Lille 1	Directeur

Table des matières

Remerciements

La rédaction d'un mémoire qui résume une grosse décennie de travaux scientifiques est l'occasion par excellence de mesurer à quel point la recherche est un processus collectif. Je voudrais donc ici exprimer ma gratitude à tous ceux qui y ont contribué directement ou indirectement, et d'avance solliciter la clémence de ceux, rares j'espère, qu'un injuste oubli aurait lésés.

Je souhaite tout d'abord adresser mes remerciements à Yves Demazeau, Amal El Fallah Seghrouchni et Jacques Ferber, qui m'ont fait l'honneur d'être rapporteurs pour ce mémoire, en dépit d'un emploi du temps lourdement chargé et d'un délai assez serré. Je suis également très reconnaissant à Rémy Courdier, Anne Ruas et Sophie Tison de l'intérêt qu'ils ont manifesté à l'égard de ce travail en acceptant de participer au jury. Quant à Philippe Mathieu, j'ai envers lui une dette considérable, pour m'avoir d'abord accueilli chaleureusement en 2002 au sein de l'équipe SMAC, puis témoigné une confiance indéfectible et amicale, que ce soit en matière de recherche ou d'enseignement. Ses conseils, ses encouragements et son énergie communicative sont pour beaucoup dans l'achèvement ce mémoire.

Mes remerciements vont également aux doctorants qui se sont lancés avec enthousiasme sur des sujets difficiles : Yoann Kubera, qui, après m'avoir forcé à préciser *ce que j'entendais par « simulation »*, a été un défenseur inexpugnable de l'approche orientée interactions et un pilier de l'équipe SMAC pendant plus de quatre ans ; et Adrien Maudet, qui a assimilé avec une rare célérité les subtilités de la cartographie et se livre avec équanimité à un numéro d'équilibriste entre résolution de problèmes et simulation orientée interactions. Quant au succès des projets Format-Store et Galaxian, il doit énormément à l'implication enthousiaste de ses participants : David Panzoli, Jean-Baptiste Leroy et Marc-Antoine Dupré, ainsi qu'aux échanges stimulants avec les collaborateurs d'Idées-3Com, notamment Dominique Ringeval.

L'évolution de notre métier vers un fonctionnement par projets censés s'enchaîner à un rythme soutenu ne doit pas nous faire oublier que c'est souvent du temps consacré à la discussion et à l'échange informel que naissent les idées les plus riches. Je suis donc très reconnaissant à Cécile Duchêne et à Guillaume Touya de s'être lancés avec moi dans une confrontation de nos problématiques. Il en résulte une coopération fructueuse et amicale, concrétisée à travers la thèse d'Adrien Maudet. De même, j'ai été très heureux des liens tissés avec des collègues biologistes, sources de questionnement et d'inspiration pour chacun, notamment François-Yves Bouget, Florence Corellou, Christian Schwartz et leurs étudiants (Observatoire océanographique de Ba-

nyuls) ainsi que Christophe Guinet et Frédéric Bailleul (Centre d'Études Biologiques de Chizé). Enfin, je souhaite souligner également la qualité des discussions qui ont lieu chaque année au sein de la communauté multi-agents francophone à l'occasion des JFSMA, et faute de pouvoir être exhaustif je mentionnerai dernièrement les échanges avec Gildas Morvan et Fabien Michel sur la simulation multi-niveaux.

Une bonne part des travaux présentés dans ce mémoire n'auraient pas vu le jour sans la participation d'étudiants motivés et dynamiques : Jérémy Scafi, Benoît Gosse, Cyril Moreau, Nicolas Petitprez, Virgile Bourse, Mickaël Delacroix, Lisa Allali, Benjamin Kubiak, Valentin Ribeiro Dubois, Laëtitia Bonte, Benoît Lacroix, Nicolas Wardaega, François Gaillard, Florent Delhaye, Ala Atrash, Thomas Clément et dernièrement Guillaume Dauster. Si certains d'entre eux se sont engagés sur la voie rocailleuse de la recherche ou de l'enseignement, j'ose espérer y avoir quelque peu contribué — et surtout qu'ils n'en éprouvent aucun regret.

Mon intégration dans le Nord tient largement aux qualités humaines d'accueil, d'écoute et d'amitié que j'ai ressenties au sein du LIFL, de l'équipe SMAC, et de l'IUT où j'ai effectué la plus grande part de mon service d'enseignement. Qu'en soient remerciés tout particulièrement (et entre autres) : Bruno Beaufils, Yann Secq, Jean-Christophe Routier, Jean-Paul Delahaye, Maxime Morge, Patricia Everaere, Bruno Mahiddine, Antoine Nongaillard, Iryna Veryzhenko, Omar Rihawi, Lisa Rougetet, Cédric Dinont, Damien Devigne, Julien Derveeuw, Marie-Hélène Verrons, François Boulier, François Lemaire, Jean-Marie Place, Isabelle Simplot-Ryl, Patricia Hecquet, Élodie Cimetière, Régine Derache, Marie-Claude et Michel Delecour, Michel Deblock, Éric Triquet, Mohammed Hamdi, Jacques Marengo, Georges Grimonprez, Sylvestre Vanuxem, Gildas Barbot, Pierre Le Brun, Max Chlebowski et Moulay-Driss Benchiboun. Je dois dire également que j'ai pris beaucoup de plaisir à enseigner au sein du département informatique de l'IUT, et plus spécialement en Licence Pro SIL DA2I où l'ambiance toujours chaleureuse des promotions m'a souvent aidé à surmonter les coups de fatigue inhérents à ce métier.

Je dois à Alexis Drogoul d'avoir fait naître en moi, il y a plus de quinze ans, un intérêt jamais démenti pour l'intelligence collective et les agents réactifs, et de m'avoir insufflé son énergie lorsqu'il dirigeait ma thèse. Mes premiers pas sur cette voie doivent également beaucoup à Jean-Pierre Briot et Anne Collinot. Je voudrais en outre évoquer le souvenir de quelques figures qui ont marqué mon parcours, d'étudiant d'abord, puis d'enseignant-chercheur, et que je qualifierais de *passeurs* : leur talent pédagogique, leur enthousiasme pour leur discipline, leur disponibilité pour des discussions à bâtons rompus m'ont ouvert des domaines entiers dont, sans eux, j'aurais continué longtemps d'ignorer les attraits, si ce n'est l'existence. J'espère n'oublier personne en citant, par ordre approximatif d'apparition dans mon environnement cognitif : Yves Jeanneret, Julien Feydy, Alain Grumbach, Jean-Louis Dessalles, Jacques Pitrat, Jean-François Perrot, Jean-Daniel Zucker et Alain Cardon.

Enfin, merci à tous ceux qui, de près ou de loin, en esprit ou par leur présence, ont accompagné mon cheminement : Dimitri, Samuel, Frédéric, David... et bien sûr ma famille, tout spécialement Anne, Esther, Yves et Marc, dont la patience et l'amour m'ont été indispensables pour mener à bien ce manuscrit.

Avant-propos

La rédaction de ce mémoire de synthèse d'habilitation à diriger des recherches a été pour moi l'occasion de prendre un peu de champ par rapport à mon quotidien d'enseignant-chercheur. Il s'agit d'abord de se retourner sur une douzaine d'années de vie professionnelle pour examiner d'un regard rétrospectif et critique les travaux poursuivis depuis ma thèse de doctorat. C'est aussi le moment de développer une certaine idée des questions scientifiques importantes du domaine, et de l'activité de recherche en général.

Après ma thèse de doctorat en informatique [50], qui s'est déroulée au LIP6 (Paris VI) de 1997 à 2001, sous la direction d'Alexis Drogoul, j'ai été recruté en septembre 2002 au sein du LIFL (Lille 1), dans l'équipe SMAC dirigée par Philippe Mathieu. Mes activités de recherche depuis cette date ont consisté à développer des méthodes et outils de simulation multi-agents permettant de modéliser efficacement et simplement des systèmes complexes à large échelle, en utilisant la notion d'interaction. Après un premier prototype de validation (plateforme PASPAS, en Prolog) j'ai d'abord travaillé sur les techniques de simulation à large échelle (projets SimuLE et MiSC), avant d'entamer à partir de 2005 une refonte complète de l'approche orientée interactions à travers le projet IODA, que Yoann Kubera dans sa thèse (2007–2010) a décliné à travers une plateforme d'expérimentation (JEDI) accompagnée d'un IDE de conception de simulations (JEDI-Builder). Le stage de François Gaillard a permis d'ajouter une couche « méta » grâce à l'explorateur LEIA (2008) qui permet de manipuler et comparer divers modèles de simulations en parallèle. Parallèlement à ces travaux, j'ai eu l'opportunité de mettre IODA en application sur des problématiques de biologie (rythmes circadiens, 2006–2007), ainsi qu'à travers un *serious game* immersif (projet Format-Store, 2010–2011) et une démonstration en réalité virtuelle (projet Galaxian, 2011–2012).

J'ai par ailleurs cherché à étendre l'approche orientée interactions en direction des simulations multi-niveaux, d'abord dans le cadre de la simulation large échelle (stage de Laetitia Bonte, 2005) puis dans le cadre de IODA (PADAWAN, 2010–2013). Ces travaux ont permis la naissance d'une collaboration avec l'IGN sur des problématiques de résolution de problèmes en cartographie, qui se traduit par la thèse d'Adrien Maudet (commencée en 2012). Enfin, il est apparu plus récemment nécessaire, pour accroître la fiabilité des simulations, de s'intéresser également à l'extraction d'infor-

mations à partir de données issues de systèmes réels, pour paramétrer judicieusement les populations d'agents (2012–2013).

Ce document est organisé comme suit. La première partie dresse un bilan des travaux menés depuis mon doctorat et qui ont été largement publiés, exploités, discutés dans la communauté SMA. Après avoir présenté le domaine et les problématiques scientifiques afférentes (chap. 1), j'aborde les techniques de « simulation orientée interactions » que j'ai contribué à développer ainsi que les recherches connexes (chap. 2) ; enfin je décris les projets applicatifs que nous avons menés au moyen de cette méthode et qui ont permis une validation empirique de notre démarche (chap. 3). Ces travaux soulèvent nombre de questions nouvelles et de pistes à explorer qui seront évoquées en conclusion de ce mémoire (chap. 6).

La seconde partie expose quant à elle des travaux en cours, c'est-à-dire commencés dans les deux à trois dernières années, et promis à des développements importants aussi bien du point de vue conceptuel que logiciel. Ainsi, la simulation multi-niveaux (chap. 4), telle que nous l'avons abordée, constitue certes une extension de la simulation orientée interactions, mais elle permet d'attaquer un large ensemble de verrous scientifiques dans la modélisation de comportements. De même, l'analyse de données à des fins de paramétrisation des simulations (chap. 5), pour laquelle nous avons mis au point des méthodes originales, s'avère un point crucial pour permettre au domaine de la simulation multi-agents de s'attaquer avec efficacité à l'étude de systèmes complexes large échelle. C'est sur la base de ces évolutions récentes que je suis en mesure de proposer un projet de recherche cohérent et ambitieux pour les années à venir (chap. 6).

On trouvera également à la fin de ce mémoire mon *Curriculum Vitae* détaillé, suivi de la liste de mes publications, des autres références bibliographiques citées et d'un index.

N.B. 1 : Par convention typographique, les expressions qui figurent en gras dans le corps du texte ont un caractère de définition.

N.B. 2 : Prenant acte d'une grande disparité d'usage dans la communauté francophone, j'ai fait le choix d'un pluriel systématique pour l'expression « simulation multi-agents » (ou « multi-niveaux »), en accord avec Grevisse [125, § 185d].

Liste des figures

Liste des tableaux

Liste des définitions

Première partie

Synthèse des travaux

The proof of the pudding is in the eating.

Proverbe anglais

Chapitre 1

Introduction : la simulation de systèmes complexes

> Entre la théorie et l'expérience, entre la formalisation mathématique et l'observation phénoménologique, la simulation offre une troisième voie : l'exploration algorithmique.
>
> Quéau [177, p. 147]

Dans ma thèse de doctorat [50], intitulée *Modèles de comportements sociaux pour les collectivités d'agents et de robots*, j'avais tout d'abord travaillé à des problématiques classiques de modélisation et de simulation, à savoir la recherche des modèles minimaux qui permettraient de reproduire certaines caractéristiques des sociétés de primates. L'objectif était, alors, de mettre en évidence des algorithmes, des comportements, des représentations qu'il serait possible de transposer dans des applications distribuées, sur des agents physiquement situés tels que des robots, pour leur permettre de gérer de façon autonome la question de leur organisation sociale. À mi-parcours, je dressais un constat d'échec de cette démarche et en appelais plutôt à l'utilisation de techniques fondées sur le développement, la sélection et l'évolution pour permettre un ancrage efficace et adaptatif des comportements de robots dans leur environnement physique. Parti de la simulation, j'avais fini par œuvrer pour les systèmes physiquement distribués...

D'une certaine façon, les années qui ont suivi ont été l'occasion d'un retour à la simulation, à ses problèmes spécifiques, à ses techniques. Je concluais mon mémoire de doctorat sur la nécessité, pour l'informaticien et plus encore pour le chercheur en IA distribuée, de donner une réalité opérationnelle à chacune des métaphores qu'il entendait employer, à travers « quelques principes généraux qui restent féconds en toutes circonstances : *parcimonie*, *incrémentalité* et démarche *expérimentale* ». Je ne pouvais être plus en phase avec l'équipe SMAC du LIFL où j'arrivais à la rentrée 2002,

puisque la mise en œuvre pratique des modèles et leur évaluation y ont toujours été l'indispensable pierre de touche de la modélisation et de la démarche théorique.

Depuis, je me suis attaché, de concert avec mes collègues, à approfondir quelques-unes des questions que soulève la simulation multi-agents, en articulant constamment la nécessité d'une méthode de modélisation facile à appréhender pour des thématiciens, la réduction des biais d'implémentation, et la mise en œuvre efficace des techniques correspondantes.

Ce chapitre introductif vise à présenter le domaine de la simulation multi-agents ainsi que les principes méthodologiques qui ont guidé mes travaux.

1.1 Présentation du domaine

Les Systèmes Multi-Agents : origines et influences

Le domaine des Systèmes Multi-Agents (SMA) est né dans les deux dernières décennies du XX^{e} siècle, avec un décollage très net depuis les années 1990. D'un point de vue informatique, on peut dire qu'il résulte de la confrontation entre d'une part les problèmes de programmation objet et de programmation distribuée, notamment les langages d'acteurs de Hewitt [128], et d'autre part les besoins de communication et de distribution pour la résolution de problème en Intelligence Artificielle, en particulier les architectures de type tableau noir de Hayes-Roth [127]. Les SMA s'inscrivent dans le cadre plus large de l'Intelligence Artificielle Distribuée (IAD) [108, 79, 201] qui englobe par ailleurs la résolution distribuée de problèmes (ou de problèmes distribués, cf. la polysémie naturelle de l'expression *Distributed Problem Solving* [110]).

Toutefois, ce domaine naissant a subi beaucoup d'autres influences, parfois de disciplines éloignées de l'informatique, qui l'ont irrigué de concepts, de métaphores, de méthodes qui leur étaient propres. Le courant de la Vie Artificielle initié par Langton [141] a par exemple entraîné une étude assez féconde des mécanismes d'auto-organisation ou de l'émergence de structures spatiales ou temporelles à partir de comportements individuels très simples. La sociologie a fourni nombre de modèles d'organisation, de représentation des normes ou de la confiance [83, 149]. Enfin, l'éthologie, science de l'étude comparée des comportements animaux dans leur milieu, constituée au cours du XX^{e} siècle mais structurée et vulgarisée principalement à travers les travaux de Lorenz [146], a exercé un ascendant indéniable, notamment en raison de son lien direct avec l'usage des SMA à des fins de simulation.

Si ce domaine a connu une croissance considérable depuis sa création, tant en matière de techniques que d'applications, il n'en garde pas moins une relative immaturité d'un point de vue scientifique. Écartelés entre des usages fort différents, qui vont de la simulation à la robotique mobile en passant par la résolution (distribuée ou non) de problèmes (eux-mêmes distribués ou non), la construction de mondes artificiels ou encore de nouvelles formes de génie logiciel [110], les SMA sont en outre pris entre les feux croisés des tenants d'une unification logicielle et de ceux qui n'y voient qu'un cadre de modélisation.

À ce jour, et en dépit de tentatives de normalisation (orchestrée notamment par la FIPA, *Foundation for Intelligent Physical Agents* de l'*IEEE Computer Society*), il n'existe

toujours pas de consensus quant à la forme logicielle que doit prendre un agent ou un système multi-agents. En effet, force est de constater que les concepts, architectures, algorithmes qui sous-tendent l'approche SMA ne se traduisent pas nécessairement par une implémentation univoque. Inversement, il n'est pas raisonnable de réduire les SMA à un simple cadre conceptuel destiné à faciliter la décomposition d'un problème : d'une part, cette approche ne se veut pas une simple méthodologie et ne possède pas non plus de formalisme théorique qui permettrait par exemple de démontrer des propriétés formelles ; d'autre part, de nombreuses propositions ont été élaborées et confrontées pour répondre à des problèmes très pratiques d'implémentation, et si une grande diversité subsiste, il s'y dessine néanmoins des schémas récurrents, en particulier pour chaque sous-domaine.

SMA, agent : définition minimale

Cette articulation entre concepts et implémentation fera l'objet de nombreuses discussions au fil de ce mémoire. Provisoirement, contentons-nous de définitions *a minima*, quitte à les enrichir en chemin.

Définition I — Système multi-agents.
Un **système multi-agents** est essentiellement un système composé *d'entités autonomes* (appelées **agents**) qui peuvent interagir les unes avec les autres au sein d'un **environnement** commun.

Cet environnement est un espace ou plus souvent une portion d'espace, continu ou discret, de topologie quelconque : il peut s'agir aussi bien d'un espace euclidien que d'une « grille », d'un graphe (par exemple pour décrire des relations d'accointance) ; il peut éventuellement se réduire à un point si l'on considère que le système n'est pas « spatialisé ». Mais son rôle premier est de servir de support à l'activité ou à la communication des agents.

Quant à **l'agent**, la manière la plus simple de le définir est celle de Russell et Norvig :

Définition II — Agent.
Un **agent** est tout ce qui peut être vu comme **percevant** son environnement au moyen de **capteurs** et **agissant** sur cet environnement au moyen **d'effecteurs** (« ***an agent*** *is anything that can be viewed as* ***perceiving*** *its environment through* ***sensors and acting*** *upon that environment through* ***effectors*** » [189]).

Ces critères peu restrictifs s'appuient uniquement sur la description de *ce que fait* l'agent, et ne fait aucune hypothèse quant aux *mécanismes internes* utilisés pour percevoir, décider et agir. La définition donnée par Ferber [110] est plus précise sur ce point (en imposant à l'agent des buts au moins implicites et des moyens pour les atteindre),

plus proche de ce fait d'une spécification opérationnelle, mais plus exigeante aussi en imposant un couplage assez fort entre les actions de l'agent et l'accomplissement de ses objectifs. Nous aurons l'occasion au chapitre 2 de revenir sur cette discussion.

La simulation multi-agents

Les travaux que j'ai menés au cours de ma thèse, et surtout depuis, s'inscrivent principalement dans un sous-domaine des SMA, la **simulation multi-agents**, en anglais *Multi-Agent Based Simulation* (MABS). Le terme de « simulation », dans le cadre informatique, recouvre en fait au moins deux pratiques complémentaires.

— La première correspond à l'acception scientifique de ce terme, et consiste à « pratiquer des tests *sur un substitut* de la situation réelle [...] (le substitut étant supposé reproduire fidèlement les caractéristiques pertinentes essentielles à la situation à caractériser) » [199], et ce afin d'évaluer la plausibilité des hypothèses qui sous-tendent un modèle. Comme dans une expérience, il s'agit donc de *contrôler* les conditions dans lesquelles se déroulent les événements à étudier, afin d'aider à l'évaluation d'un modèle issu d'une autre discipline scientifique.
— La seconde correspond à l'usage courant du terme « simulation » : l'animation d'un monde artificiel, proche ou non du monde réel, et dont l'usage peut être ludique (jeu vidéo, cinéma), pédagogique (simulateur de conduite, *Serious Game*), artistique [132]...

En pratique, les objectifs visés dans chacune de ces deux grandes catégories peuvent être assez diversifiés : prédiction de l'évolution d'un système, explication de ses mécanismes, découverte de phénomènes inattendus, preuve, entraînement, divertissement, remplacement d'acteurs humains, etc. [58].

Cela nous amène à proposer la définition suivante pour les simulations multi-agents :

Définition III — Simulation multi-agents.
La simulation multi-agents est le domaine qui étudie les méthodes et algorithmes, dans le cadre conceptuel et opérationnel des SMA, qui permettent de construire des techniques de représentation et de production des connaissances ainsi que de calcul d'événements, à la fois :

— **fiables**, c'est-à-dire sans biais introduits dans les résultats par des choix implicites de modélisation ou d'implémentation ;
— **efficaces** en termes de temps d'exécution ou de volume d'information pouvant être traité ;
— **manipulables** autant que possible par des thématiciens (experts non informaticiens du domaine auquel s'applique la simulation visée) qui doivent pouvoir paramétrer voire programmer eux-mêmes les outils de simulation ;
— **intelligibles** en ce qui concerne la représentation du modèle et l'interprétation des résultats de simulation par ces mêmes thématiciens ;
— **modulaires** pour permettre une révision aisée de parties du modèle ;

— **polyvalentes**, autrement dit permettant de traiter de la même façon non seulement des simulations scientifiques appliquées à des domaines variés, mais également des simulations destinées à produire des mondes artificiels.

L'une des difficultés de la conception d'une simulation réside dans le fait qu'elle suit un processus complexe au cours duquel le modèle du phénomène établi par le thématicien, dit **« modèle thématique »** (parfois appelé « modèle non formel »), est d'abord reformulé dans le cadre d'un formalisme donné, pour constituer un **« modèle conceptuel »** (ou « modèle formel ») [191, 182]. Celui-ci est à son tour transformé en un **« modèle computationnel »** (ou « modèle exécutable », ou encore « modèle opérationnel ») [106, 99] qui comporte les spécifications nécessaires à l'implémentation. Enfin ce dernier est traduit en pratique en un programme informatique au moyen d'un langage de programmation ou au sein d'une plateforme de simulation. Chacune de ces étapes est susceptible de déroger aux exigences que nous avons formulées ci-dessus [105, 115].

Domaines proches

On trouvera également dans la littérature la terminologie « *Individual-Based Modeling* » (IBM) ou « *Agent-Based Modeling* » (ABM) [130, 68]. Le recouvrement avec la simulation multi-agents est très large ; toutefois l'accent y est en général moins mis sur la notion de collectif (et d'interactions) que sur l'individuation des entités qui composent le système. Ces domaines sont issus majoritairement de la sociologie et de l'écologie, où l'idée même de descendre au niveau de granularité des individus pour formuler des hypothèses qui se prêtent à l'expérience est relativement récente, les dénominations IBM et ABM visant précisément à marquer cette spécificité par rapport à l'approche dominante. En pratique, on constate que la différence avec la simulation *multi-agents* s'amenuise constamment. Ainsi par exemple, Craig Reynolds maintient une liste de simulations mettant en œuvre des « *Individual-Based Models* » qui sont pour une très vaste majorité des modèles multi-agents (`http://www.red3d.com/cwr/ibm.html`).

1.2 La simulation multi-agents de systèmes complexes

En raison de sa capacité d'une part à fournir des « laboratoires virtuels » qui manipulent des entités informatiques isomorphes à celles des thématiciens, et d'autre part à évaluer des modèles explicatifs et non seulement prédictifs, la simulation multi-agents a progressé de façon constante et surtout couvre des domaines sans cesse plus variés.

De ses domaines applicatifs de prédilection, comme l'éthologie [23, 98, 205], la sociologie [92, 121], le trafic routier [109] ou l'économie [184], la simulation multi-agents s'attaque désormais à la biologie systémique [183], aux systèmes physiques ou chimiques [207, 210], aux écosystèmes [84, 74], aux processus de diffusion [112], à la

finance [57, 75], aux phénomènes de foule [68, 165, 196], aux réseaux sociaux [181]. Ces domaines d'application récents présentent des caractéristiques communes qui soulèvent des difficultés tout à fait nouvelles pour les SMA :

— La modélisation y est dominée très largement par l'approche « systèmes dynamiques », c'est-à-dire une approximation continue basée sur des systèmes d'équations différentielles, parfois tempérée par des méthodes stochastiques : les SMA ont donc en général à y *faire la preuve de leur intérêt*, au contraire des domaines « historiques ».
— Cette prédominance du continu vient évidemment du fait qu'on se trouve face à un *très grand nombre d'entités plus ou moins interchangeables* : l'hypothèse que ces entités sont identiques dans leur comportement général (en dépit de possibles variations de leurs propriétés), même si elle n'est pas vérifiée en permanence, constitue souvent une assez bonne approximation.
— Le recours à la simulation multi-agents vient souvent du besoin de gagner en *intelligibilité* et en *flexibilité* pour pouvoir à la fois tenir un discours explicatif à partir d'un modèle qui rend compte correctement des observations, et le modifier facilement lorsque d'autres hypothèses doivent être testées.
— Les comportements individuels, lorsqu'ils se différencient, sont mal connnus car il est plus difficile d'observer les entités du système que lorsqu'il s'agit de « gros » individus comme en éthologie : les méthodes d'acquisition de ces informations sont souvent statistiques ou destructives (broyage de cellules par exemple).
— Enfin, ces nouveaux domaines sont producteurs de *quantités de données colossales*, notamment en raison des méthodes de mesure automatique qu'elles ont développées durant les dernières décennies, confortées par l'accroissement des capacités de traitement et de stockage numériques.

Systèmes complexes, systèmes large échelle

En fait, si l'on considère ces nouveaux domaines, on constate qu'ils se situent à l'intersection de deux ensembles de phénomènes :

— D'un côté, on trouve les **systèmes complexes**, constitués d'entités qui interagissent les unes avec les autres de façon locale et non triviale (non linéaire notamment, ou avec des boucles de rétroaction), de sorte que l'évolution du système ne peut être prédite par l'intégration pure et simple des lois qui régissent les entités : la prédiction nécessite une expérimentation ou une simulation. Un sytème complexe peut être composé de très peu d'entités, par exemple le fameux problème gravitationnel des trois corps pour lequel il existe bien une solution analytique formulée par Sundman [200], mais qui ne permet de tirer aucune prédiction de long terme sur les trajectoires des corps et leur stabilité.
— L'autre versant est constitué par les **systèmes large échelle** qui impliquent un *très grand nombre d'agents*, appartenant à des *familles variées* et liées entre elles par une *grande diversité d'interactions*. Un système large échelle peut être soumis à une évolution macroscopique déterministe et formulable de façon simple, comme par exemple dans la théorie cinétique des gaz de Boltzmann [67].

La conjonction de ces deux groupes de systèmes complique considérablement la tâche du modélisateur : les systèmes large échelle ne peuvent plus passer par des méthodes statistiques ou équationnelles dès lors que les interactions entre les entités qui les composent ne sont plus triviales ; mais réciproquement, les approches plus « qualitatives » (théorie des systèmes [66], cybernétique [214], systèmes multi-agents, etc.) qui ont aidé à comprendre les mécanismes à l'œuvre dans les systèmes complexes, doivent se doter de nouvelles armes pour faire face à *l'effet de masse* du passage au large échelle.

Pour ce qui est des systèmes multi-agents, originellement appliqués à la simulation ou à la production de systèmes complexes, le principal problème serait donc de passer au large-échelle. Mais il ne s'agit pas seulement d'un changement de volume, dont viendrait aisément à bout l'accroissement régulier des capacités de calcul des machines : il faut également revoir la façon de représenter les connaissances pour rendre celles-ci plus *lisibles*, permettre facilement la *révision* de modèles dont l'expression est de plus en plus compliquée, et de façon générale *faciliter et simplifier* l'accès du thématicien au modèle conceptuel.

Nous verrons dans le prochain chapitre comment l'appproche orientée interactions cherche à répondre à ces préoccupations.

1.3 Questions méthodologiques

> Il n'est pas sûr que la nature soit simple. Pouvons-nous sans danger faire comme si elle l'était ?
>
> Poincaré [175, p. 173]

Durant ces travaux, je me suis appuyé sur quelques principes épistémologiques que je considère comme majeurs.

Le principe de parcimonie

Le premier d'entre eux est le **principe de parcimonie** ou « rasoir d'Ockham », selon lequel on se doit de ne pas multiplier les entités plus que nécessaire. Ce principe qui s'affichait au fronton du temple de Delphes (sous la forme Μηδὲν ἄγαν : « rien de trop ») est devenu un des fondements des sciences empiriques modernes, et conduit à privilégier les modèles qui font appel au nombre d'hypothèses le plus faible, et à aussi peu de concepts que possible, ces concepts se devant en contrepartie d'offrir un champ d'application étendu. Comme le rappelait Poincaré [175], « si nous construisons une théorie fondée sur des hypothèses multiples, et, si l'expérience la condamne, quelle est parmi nos prémisses celle qu'il est nécessaire de changer ? Il sera impossible de le savoir. Et inversement, si l'expérience réussit, croira-t-on avoir vérifié toutes ces hypothèses à la fois ? ».

Or le passage à la simulation informatique introduit mécaniquement de nombreuses questions qui ne se posaient pas nécessairement au thématicien mais que l'informaticien *doit* résoudre : le modèle conceptuel et le modèle computationnel définissent en effet les propriétés du substrat numérique qui vient se substituer à la situation réelle, et c'est donc sur les choix qui y sont faits en matière de représentation des connaissances, d'algorithmique et d'implémentation, que repose la fiabilité de l'expérience virtuelle, autrement dit sa capacité à « reproduire fidèlement les caractéristiques pertinentes essentielles à la situation à caractériser » [199]. Bien évidemment le principe de parcimonie s'applique aussi bien aux hypothèses des thématiciens qu'au processus d'élaboration des modèles conceptuels et computationnels.

La parcimonie se pose ainsi en indispensable garant de l'intelligibilité et de la fiabilité de nos systèmes artificiels : car, lorsque l'on cherche à reproduire un phénomène collectif intéressant à partir du comportement des entités qui le composent, la moindre des choses est de pouvoir expliquer très précisément ce que font chacune de ces entités prises individuellement et comment le calcul de leurs interactions est mené à bien. Ce principe nous conduit donc à rechercher des *modèles et méta-modèles minimaux* et à viser une *reproductibilité* parfaite des expériences. Il se traduit aussi par une volonté *d'unifier des concepts*, quand c'est possible, par une généralisation pertinente (comme nous le verrons pour la notion d'agent plus loin).

L'explicitation des hypothèses

Pour pouvoir appliquer le principe de parcimonie aux modèles thématiques (et par suite, conceptuels), il faut être en mesure *d'expliciter* toutes les hypothèses d'un modèle, ce qui ne va certainement pas de soi. Les thématiciens élaborent leurs hypothèses dans un cadre théorique qui leur est propre, et comporte un grand nombre d'éléments implicites ; lorsque l'informaticien pense transcrire ces modèles sous une forme computationnelle, il s'expose non seulement à oublier ces présupposés qui n'ont pas pour lui de caractère d'évidence, mais en outre à introduire dans sa propre construction des choix liés au type de modèle qu'il conçoit, à l'implémentation qu'il en donne, voire à l'architecture de calcul qu'il utilise. Or, il apparaît que les simulations multi-agents sont tout particulièrement exposées à ce travers : par la métaphore de l'individu et de l'environnement, elles peuvent aisément prêter le flanc à *l'illusion d'une transposition immédiate* d'un concept du modèle thématique en un concept du modèle conceptuel ou même du modèle computationnel [105, 134].

L'un des efforts de la modélisation consiste donc, d'une part, à *forcer le thématicien à expliciter certaines de ses hypothèses tacites* ; d'autre part, à lui fournir un cadre de simulation dans lequel la plupart des choix possibles de représentation des connaissances et d'implémentation ont été *répertoriés* et où *des choix par défaut lui sont explicitement proposés*.

Une démarche constructionniste

Le second principe directeur de mes travaux est le **constructionnisme**, pris dans un sens quasiment mathématique, à savoir que les modèles et les algorithmes que nous proposons n'ont pas de valeur tant qu'ils n'ont pas fait l'objet d'une *construction*

qui les fait exister, autrement dit d'un programme que l'on peut traiter par l'étude expérimentale et juger à l'aune de ses qualités logicielles. Dans la mesure en effet où l'IA distribuée ne relève pas de l'informatique théorique et qu'on peut difficilement y établir des preuves formelles, il nous semble qu'une *validation empirique* (qui peut d'ailleurs prendre des formes variées) est la seule démarche applicable à notre domaine.

D'une certaine façon, on peut considérer que, de même l'expérience ou la simulation permettent de corroborer ou de réfuter les hypothèses du modèle thématique, de même la mise en œuvre pratique d'une méthodologie ou d'une approche de simulation, à travers des plateformes logicielles et des cas d'étude concrets, constitue une expérimentation toujours renouvelée de leurs avantages théoriques, et sans aucun doute profitable car elle contribue ainsi à renforcer leur scientificité. C'est une forme de **réfutabilité** [176] de la démarche théorique, peut-être la seule dont nous puissions véritablement faire état dans notre discipline.

Ce constructionnisme peut s'entendre également dans un sens proche de celui défendu par Papert [170], en ce qui concerne la façon dont nous apprenons à manipuler des outils notamment informatiques. En effet, la simulation multi-agents a vocation à modéliser des problèmes issus de domaines dont les thématiciens sont en général peu formés à la programmation, ou même seulement à l'algorithmique. La conception d'un modèle de simulation doit leur permettre de procéder par essais et erreurs, et ce à moindres frais du point de vue de l'écriture de code : autrement dit, elle doit être **incrémentale**. Nous nous devons donc de construire nos outils de telle sorte que les thématiciens n'aient plus, si possible, qu'à assembler correctement les bonnes pièces, sans avoir à se préoccuper de leur fonctionnement interne.

Chapitre 2

La simulation orientée interactions : principes et outils

> Une méthode excellente finit par perdre sa fécondité si on ne renouvelle pas son objet.
>
> Bachelard [59, p. 14]

Lors des deux dernières décennies, la simulation multi-agents n'a cessé d'étendre son emprise sur des domaines toujours plus nombreux, et ce me semble-t-il précisément en raison de sa capacité à rendre compte des causes des phénomènes et pas seulement à prédire leur évolution sous certaines circonstances. Sous l'influence déterminante de l'éthologie, cette discipline a opérationnalisé les concepts d'individu, de groupe, de comportement, de réflexe, d'environnement d'une façon particulièrement féconde. L'idée prédominante de la dernière décennie du siècle passé était d'ailleurs que la simulation de systèmes sociaux naturels, des insectes aux hommes en passant par d'autres primates, allait fournir de nouvelles méthodes informatiques pour organiser des sociétés d'agents artificiels ou résoudre des problèmes collectifs. J'ai moi-même souscrit à l'optimisme de ces recherches avant d'en reconnaître les limites [50].

Certes, la simulation de sociétés naturelles a permis de synthétiser des points de vue qui s'exprimaient de façon éparse dans diverses disciplines, et d'illustrer à travers de nombreux exemples convaincants ces mécanismes qui font apparaître des phénomènes complexes à partir d'acteurs simples. Mais, hélas ! hormis un ensemble de mécanismes, maintenant assez bien connus (rétroactions négatives mais surtout positives, ordre par le bruit, brisure de symétrie, stigmergie, localité des perceptions et des actions tant dans l'espace que dans le temps, effets de seuil et de masse critique) qui somme toute avaient déjà été identifiés par la cybernétique et par l'étude des systèmes dissipatifs, et à l'exception de quelques techniques comme le *flocking* [180] ou

le routage « fourmis » [93], force est de reconnaître que n'a pas vraiment conduit à une révolution algorithmique.

Aussi, une divergence s'est-elle progressivement installée entre le domaine de la simulation multi-agents et celui de la conception d'applications informatiques utilisant les SMA. Par simulation, nous devons entendre la création de programmes destinés à reproduire des phénomènes, que ce soit à visée scientifique (évaluation d'hypothèses, prédiction, aide à la décision), ludique (cinéma, jeux vidéo), pédagogique (formation, entraînement, sensibilisation) ou artistique. Seuls les objectifs de ces simulations diffèrent, mais nous verrons qu'il est possible, pour les réaliser, d'employer les mêmes procédés. Dans ce chapitre, nous allons donc voir quel genre d'approche spécifique à la simulation nous avons choisi de construire, et nous reviendrons dans le chap. 6 sur la question de la conception d'applications multi-agents.

2.1 Contexte scientifique

Une focalisation excessive sur l'individu

Dans les domaines d'application visés à l'origine par les simulation multi-agents, tels que l'éthologie [205, 98, 23], la sociologie [92, 121], le trafic routier [109] ou l'économie [184], *la notion d'individu est centrale*. Certes, l'individu n'existe pas en tant que tel, isolé, mais en tant que membre d'un groupe ou du moins comme entité en relation avec d'autres. Néanmoins, c'est bien sur l'individu que porte l'effort de modélisation : ses propriétés, ses représentations mentales (numériques ou symboliques), ses buts implicites ou explicites, et... son comportement.

Par ailleurs, force est de constater que l'approche multi-agents est née est s'est développée dans le contexte historique de la programmation orientée objets. Nombre de ses inventeurs viennent du monde des objets des acteurs ou des composants [117, 218, 129], et l'on sent très tôt la nécessité de définir une frontière entre agents et objets [110]. Réciproquement, il est évidemment pratique de programmer un SMA dans un langage orienté objets en faisant dériver d'un même ancêtre les diverses familles d'agents nécessaires au modèle.

Une des conséquences de cette origine commune est sans doute la prédominance de la *structure* (l'entité agent) sur la *fonction* (le comportement attribué à l'agent). On constate en effet que dans les architectures classiques d'agent, le comportement est vu comme une fonction propre à l'agent, codée en tant que méthode (au pire) ou composant logiciel (au mieux) *de l'agent*. La conception d'un modèle de simulation passe par l'identification des agents, *puis* par la définition du comportement de chacun d'entre eux. L'implémentation commence elle aussi par les agents, puis s'intéresse à leurs comportements.

Cette sujétion du comportement vis-à-vis de l'agent est, selon nous, source de nombreux problèmes, parmis lesquels :

— le *manque de réutilisabilité des développements* d'une simulation à une autre : il est difficile d'affecter un comportement à un agent autre que celui pour lequel il a été écrit, car le comportement développé s'appuie sur les spécificités de l'agent dont il provient ;

— le *manque de flexibilité des modèles de simulation* : si un agent a été caractérisé par un ensemble de comportements, il est parfois difficile de lui en ajouter ou de lui en retirer sans compromettre la cohérence de l'ensemble ; *a fortiori* si des « objets » dépourvus de comportements mais utilisables comme ressources ont été introduits en tant que tels dans la simulation, il est compliqué de les doter *a posteriori* d'un comportement qui correspondrait à une évolution nécessaire du modèle ;
— le *caractère fortement procédural ou algorithmique du comportement*, qui ne va pas forcément de soi pour des thématiciens, souvent plus habitués à une formulation à base de règles ;
— dans le pire des cas, une *perte complète de l'intelligibilité des comportements* lorsque ceux-ci sont construits à partir d'architectures numériques (réseaux de neurones, algorithmes génétiques, etc.) — heureusement, cela concerne moins la simulation que la conception d'applications SMA ;
— le risque de perdre une partie de la connaissance initiale lors du passage au modèle computationnel, car des similarités connues dans les comportements effectués par des agents de familles différentes peuvent se trouver masquées précisément par l'encapsulation de ces comportements au sein des agents.

Séparer le déclaratif du procédural

Cette dépendance du comportement vis-à-vis de l'agent est d'autant plus surprenante que parmi les disciplines qui ont influencé les SMA, on en compte au moins deux qui auraient pu conduire à des orientations radicalement différentes. En premier lieu, l'éthologie [146], qui a fortement imprégné les méthodologies et les architectures, identifie de grandes *familles de comportements* indépendamment des variantes dont leur exécution peut faire l'objet : ces comportements « génériques » sont caractérisés par leur *fonction* (alimentation, reproduction, défense du territoire, etc.). Ils présentent un caractère **téléonomique** [161], c'est-à-dire qu'ils sont dotés d'une *finalité implicite* qui a été construite par l'évolution et rend l'ensemble de leur comportement aussi cohérent que s'ils planifiaient leurs actions pour atteindre un but explicite. Cette caractérisation fonctionnelle est à la base même de la conception de comportements dans une architecture d'agent réactif [110], mais sa mise en œuvre ne pousse pas au maximum la logique sous-jacente d'une forme d'*universalité* des comportements.

De même les systèmes experts, dont sont dérivés les architectures de tableaux noirs ancêtres des SMA, doivent leur succès passé à la mise en œuvre efficace d'une *séparation entre les aspects déclaratifs* (connaissances constituées de bases de faits et de bases de règles) *et les aspects procéduraux* (moteurs de chaînage avant, arrière ou mixte) de leur conception et de leur implémentation.

De fait, cette séparation existe dans l'architecture d'agent BDI (*Belief, Desire, Intention*) de Rao et Georgeff [178], mais celle-ci impose une modélisation d'un haut degré cognitif qui, d'une part, est impraticable pour bon nombre de thématiciens des domaines couramment visés par la simulation multi-agents, et d'autre part, suppose des représentations et des inférences dont la complexité ne permet pas de montée en charge.

Une première grande orientation de notre démarche a donc été de tenter de remédier à ces problèmes. Dans le même temps, nous avons dû garder à l'esprit que les

domaines d'application nouveaux que visait la simulation multi-agents étaient non pas seulement des systèmes complexes, mais aussi des systèmes large échelle.

2.2 L'approche orientée interactions (IODA)

> **interaction :** action de deux ou plusieurs objets l'un sur l'autre.
>
> LITTRÉ [144]

☞ `http://www.lifl.fr/SMAC/projects/ioda/`

L'idée d'utiliser explicitement des règles impliquant plusieurs agents pour décrire de façon abstraite leurs comportements est née dans l'équipe SMAC de Lille au début des années 2000 [150]. C'est la base même de l'approche orientée interaction que l'on peut définir comme suit :

Définition IV — APPROCHE ORIENTÉE INTERACTIONS.
L'approche orienté interactions repose sur les points suivants :

1. *toute entité* susceptible de jouer un rôle dans le modèle du phénomène à simuler est représentée par *un agent* ;
2. *tout comportement* réalisable par des agents est décrit de façon abstraite (c'est-à-dire en exprimant ce qu'il a de général) sous la forme *d'une règle* appelée **interaction** ;
3. le *moteur de simulation* est totalement indépendant des agents et des interactions, il ne fait qu'appliquer des algorithmes qui déterminent quelles interactions sont exécutées, et par quels agents.

L'une des caractéristiques majeures qu'il faut souligner ici est bien la volonté *d'unification* (et par conséquent, de simplification) des concepts utilisés ordinairement dans un SMA. Contrairement à l'approche canonique où les « agents » véritables (i.e. les entités dotées d'un « comportement ») cohabitent avec des « objets » ou « ressources », voire avec des « artefacts », des « objets actifs » ou autres hybrides, dans l'approche orientée interactions **tout est agent** [39], au sens où toute entité pertinente pour la modélisation est représentée dans le simulateur par un agent, et ce quels que soient le degré et la complexité des actions où cette entité est impliquée. Nous reviendrons sur ce choix plus loin (p. **??**).

Cette position du tout-agent ne concerne que les entités du modèle : l'environnement, quant à lui, est d'abord un *espace* contenant ces entités (cf. p. 5) et à ce titre, n'a

pas vocation à être agentifié. Nous verrons ultérieurement que cette distinction entre entités et espace permet également de donner une cohérence unificatrice à l'approche multi-niveaux que nous défendons (ch. 4).

Une telle démarche vise à séparer au maximum les aspects déclaratifs des aspects procéduraux d'un modèle de simulation multi-agents. Elle s'est déclinée au sein de l'équipe SMAC à travers deux axes principaux :

— la simulation de comportements pour agents rationnels situés, à travers le projet CoCoA (pour *COgnitive COllaborative Agents*) dirigé par Jean-Christophe Routier, qui vise notamment à doter les personnages virtuels de jeux vidéo de capacités à raisonner sur leurs buts de façon réaliste, en tenant compte par exemple d'une perception limitée, de traits de personnalité et de planification de comportements d'équipe [90, 186, 103] ;

— la simulation d'agents situés plutôt réactifs dans des systèmes large échelle, c'est-à-dire faisant intervenir de grandes quantités d'agents et de familles d'agents, chacune susceptible d'interagir de nombreuses façons avec beaucoup d'autres : dans la suite, je me focaliserai principalement sur ce dernier axe qui constitue le cœur de mes travaux depuis 2002 (date de mon arrivée dans l'équipe SMAC).

Les pièges du « large échelle »

Les premiers efforts dans ce sens nous ont beaucoup appris, au sens où ils ont permis de découvrir des impasses et d'élaborer les moyens d'en sortir.

Durant mes premières années dans l'équipe SMAC (2002–2005), je me suis focalisé sur les procédés techniques permettant d'aborder des simulations large échelle au moyen d'un SMA, ce qui a conduit à développer la plateforme de simulation SimuLE (pour « SIMUlation Large Echelle »), permettant de faire cohabiter plusieurs dizaines de milliers d'agents, disposant vues paramétrables, de *workflows* de traitement des observables, etc. (fig. 2.1) et donnant une première implémentation de la notion d'interaction [33, 45].

Plusieurs étudiants ont participé à ce projet, soit durant leur stage de DEA (Benoît Gosse en 2004, Laetitia Bonte en 2005), soit à l'occasion d'un projet de DESS (Mickael Delacroix et Virgile Bourse du DESS IAGL en 2005) ou d'ingénieur (Nicolas Petitprez et Cyril Moreau, ainsi que Benjamin Kubiak et Valentin Ribeiro Dubois de Polytech'Lille en 2005 et 2006 respectivement), ou d'autres stages (Lisa Allali, 2^{e} année, ENS).

Toutefois, il est apparu au fur et à mesure de l'élaboration de cette plateforme qu'il ne servait à rien de pouvoir monter à l'échelle en termes de nombre d'agents sans disposer en premier lieu d'un cadre approprié pour *élaborer* les modèles conceptuels sous-jacents, pour *discuter* avec les experts et pour *analyser* les comportements observés. Les choix de conception effectués restaient encore fortement teintés de l'approche multi-agents « classique » et, si la simulation elle-même offrait des performances intéressantes, la conception d'un modèle et son analyse présentaient parfois une complexité proche de celle du phénomène ciblé !

Les travaux sur les systèmes à large échelle se sont également concrétisés à travers la plateforme MiSC, dédiée à l'étude des modèles de temps et d'espace pour la simulation de foules [31, 44].

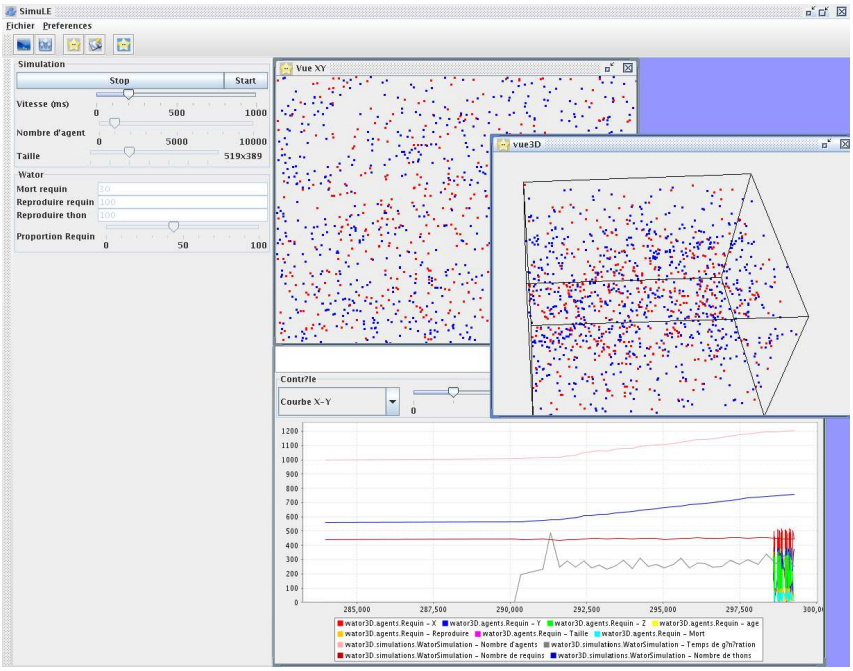


FIGURE 2.1 – Une vue de la plateforme SimuLE.

Néanmoins c'est principalement à partir d'une « déconstruction » méthodique des points problématiques rencontrés lors de la mise au point de la plateforme SimuLE que nous avons pu élaborer l'approche orientée interactions sous sa forme actuelle. Nous nous sommes également appuyés sur un autre prototype que je ne fais que mentionner, la plateforme PASPAS (stage de Jérémy Scafi, Polytech'Lille, 2003), dotée d'un moteur en Prolog, qui permettait d'approfondir l'usage de règles pour la description de comportements.

Le projet IODA

Le **projet IODA** (pour : *Interaction-Oriented Design of Agent simulations*) a pris naissance courant 2005 et vise à définir un cadre conceptuel de description des comportements des agents sous la forme d'interactions, mais également une méthodologie de conception de simulations et des algorithmes permettant d'une part d'implémenter un moteur de simulation efficace, fiable et hautement configurable, et d'autre part d'automatiser une bonne part du passage du modèle conceptuel au code.

Dans la mesure où ces travaux ont déjà fait l'objet de publications nombreuses, je me contenterai dans ce qui suit d'en retracer les grandes lignes, en renvoyant aux articles correspondants pour de plus amples détails, plus particulièrement à l'articles de synthèse de 2011 [2] et à la thèse de doctorat particulièrement roborative de KUBERA [134].

2.2.1 Un cadre d'expression des connaissances

Dans les nouveaux domaines d'application de la simulation multi-agents, l'individu n'est plus comme on l'a vu au centre des préoccupations des thématiciens. Bien

souvent, les hypothèses portent non pas sur les caractéristiques des entités (car ces caractéristiques sont soit connues de longue date, soit mesurables statistiquement, soit du ressort d'une autre discipline, soit pratiquement inconnaissables), mais bien sur *les relations* que ces entités entretiennent les unes avec les autres. Autrement dit, il s'agit de déterminer quelles sont les entités qui assurent chacune des *fonctions* du système.

Aussi, il est indispensable de pouvoir exprimer explicitement les hypothèses du modèle en séparant au maximum les aspect procéduraux des aspects déclaratifs, et de séparer en outre dans le comportement ce qui est générique (son « essence », ce qui fait qu'on peut *identifier* ce comportement sous un nom univoque) de ce qui est propre à l'agent (les *modalités* du comportement qui résultent des capacités élémentaires de perception et d'action des agents qui y participent). Pour y parvenir, nous avons été amenés à donner une réalité opérationnelle à la notion d'interaction.

Tout comportement est une interaction

La notion *d'interaction* est utilisée depuis longtemps dans la terminologie des SMA, par exemple pour décrire une situation dans laquelle deux agents agissent l'un sur l'autre dans une longue durée [110] (c'est une définition « éthologisante » de l'interaction [152, article « Interactions entre animaux »]), ou comme élément de méthodologie d'analyse et de conception [88]. Nous avons néanmoins choisi de conserver le terme, tout en lui redonnant une connotation plus physique (action mutuelle de deux corps l'un sur l'autre) :

Définition V — Interaction.
Une **interaction** est une *règle conditions/actions* impliquant plusieurs agents. L'un des agents est dit **source** de l'interaction : c'est lui qui **effectue** l'interaction ; les autres agents, qui **subissent** l'interaction, sont appelés **cibles**.

Une interaction est composée :

— d'un **déclencheur** (*trigger*) qui exprime les motivations implicites ou les buts explicites de l'interactions ;

— d'une **condition** déclenchement qui vérifie des prérequis physiques ou logiques nécessaires à l'accomplissement des actions ;

— d'une séquence **d'actions** affecter les agents participants et leur environnement.

Le but d'une telle règle est de donner une *forme cohérente* à une séquence d'actions impliquant plusieurs agents, de façon à vérifier la définition d'une « interaction forte » (*strong interaction*) au sens de Michel *et al.* [157]. Pour préserver l'autonomie des agents et le caractère générique de l'interaction, il est exclu de faire appel dans le déclencheur, la condition ou les actions à des propriétés des agents amenés à y participer. L'expression de ces trois composantes doit donc passer par des fonctions et procédures *abstraites* : des **primitives** (fig. 2.2).

Le seul prérequis pour qu'un agent puisse être source ou cible d'une interaction, c'est donc qu'il possède les primitives nécessaires et leur donne sa propre implémen-

L'interaction Manger

```
INTERACTION : manger(Source, Cible)
DECLENCHEUR : Source.A_FAIM()
CONDITION   : Cible.BON_ETAT()
ACTIONS     : Source.ASSIMILER(Cible), Cible.DETRUIRE()
```

FIGURE 2.2 – Un exemple d'interaction et de primitives. L'interaction Manger est réalisable entre un agent source et un agent cible si, 1° il existe une motivation pour cela (l'agent source doit « avoir faim » : primitive `A_FAIM` de la source), 2° les conditions sont vérifiées (l'agent cible doit être en bon état, mûr, ou simplement comestible : primitive `BON_ETAT` de la cible). Dans ce cas les actions peuvent être effectuées : d'abord la source utilise sa cible comme source d'énergie par exemple (primitive `ASSIMILE` de la source), puis la cible est retirée de la simulation (primitive `DETRUIRE` de la cible).

tation. Du point de vue de la modélisation, ces primitives peuvent être vues comme les *capacités élémentaires de perception et d'action des agents*.

Le principe général pour la réalisation d'une simulation orientée interactions est donc de doter les agents de la possibilité *d'effectuer* ou de *subir* certaines interactions avec d'autres agents, puis de les placer dans un même environnement. Schématiquement, une interaction se déclenche dès qu'un agent pouvant en être la *source* et un agent pouvant en être la *cible* sont proches et qu'ils vérifient les déclencheurs et conditions de l'interaction (fig. 2.3).

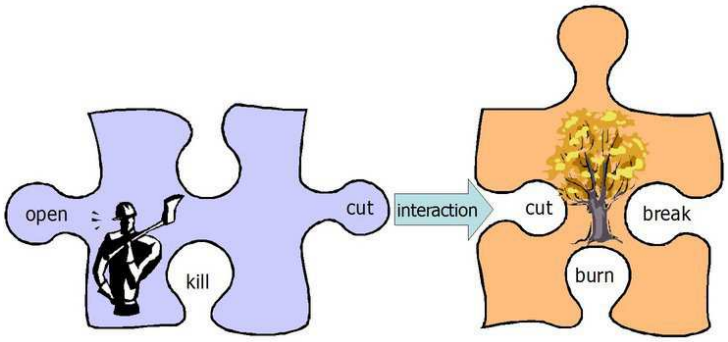


FIGURE 2.3 – Principe général d'une simulation orientée interactions. Deux agents sont susceptibles de participer à une interaction (ici, Cut) si l'un est capable *d'effectuer* cette interaction (ici, le bûcheron) et l'autre (ici, l'arbre) capable de la *subir* (emprunté à ROUTIER [186]).

Cela suppose d'avoir préalablement défini quels agents pouvaient interagir entre eux et avec quelles interactions.

Qui fait quoi, avec qui : la matrice d'interaction

Pour spécifier quelles sont les familles d'agents qui peuvent interagir, et selon quelles modalités, nous avons opté pour une représentation matricielle. Chaque ligne correspond à une famille d'agents pouvant jouer un rôle en tant que source, chaque colonne à une famille pouvant jouer un rôle en tant que cible. La **matrice d'interac-**

tion qui en résulte permet donc de visualiser très clairement qui peut faire quoi, et avec qui (tab. 2.1). Chaque membre d'une famille présente sur une ligne peut effectuer certaines interactions avec les membres des familles de chaque colonne.

Cette représentation matricielle permet de spécifier également une *relation d'ordre* entre les interactions réalisables, du point de vue de l'agent source, au moyen d'une *priorité*. On peut aussi imposer à l'interaction de ne se dérouler que dans un rayon limité en indiquant une *garde de distance*. C'est à partir de cette matrice que le moteur de sélection d'interaction (cf. §2.2.3) détermine quelle interaction un agent va réellement effectuer et avec quelle(s) cible(s).

Cibles / Sources	∅	Arbre	Herbe	Mouton	Chèvre	Loup
Arbre						
Herbe	Pousser (0) SePropager (1)					
Mouton	Mourir (3) Errer (0)		Manger (2 ; 0)	Procréer (1 ; 0)		
Chèvre	Mourir (4) Errer (0)	Grimper (3 ; 1)	Manger (2 ; 0)		Procréer (1 ; 0)	
Loup	Mourir (4) Errer (0)			Manger (3 ; 1)	Manger (2 ; 1)	Procréer (1 ; 0)

TABLE 2.1 – Un exemple de matrice d'interaction, appliqué à un écosystème-jouet. Chaque case contient les interactions réalisables entre familles d'agents, la source étant donnée par la ligne et la cible par la colonne. On notera la colonne ∅ qui représente les interactions *réflexives*, i.e. pour lesquelles c'est l'agent source lui-même qui sert de cible (p. ex. les déplacements ou la mort). Par ailleurs chaque interaction est accompagnée d'un *niveau de priorité* (premier nombre) relativement aux autres interactions réalisables par la même source, et d'une *garde de distance* pour les interactions non réflexives (second nombre) qui donne la distance maximale entre les agents pour que l'interaction puisse avoir lieu. Dans cet exemple, un agent Loup peut effectuer l'interaction Manger sur un agent Chèvre, ou encore Procréer avec un autre agent Loup ; il peut également effectuer Errer si aucune autre interaction n'est réalisable.

Certaines interactions sont effectuées de façon *réflexive* : cela correspond à des situations dans lesquelles la cible de l'interaction est en fait la source elle-même (colonne notée ∅). En général il s'agit de situations où la règle sous-jacente ne concerne qu'un seul agent, mais par souci d'uniformité elles sont représentées de la même façon que les règles impliquant des agents différents.

L'intérêt de cette représentation est qu'elle permet de spécifier des modèles de façon très visuelle, en quelque sorte par *glisser-déposer* comme c'est le cas dans la démonstration de la plateforme JEDI (cf. §2.3.1). Pour le thématicien, une fois définies les familles d'agents et les interactions pertinentes pour une classe de phénomènes donnée, il ne reste plus qu'à placer des interactions dans cette matrice, ce qui peut être fait de façon incrémentale en testant la simulation après chaque modification.

On pourrait objecter que l'existence d'une telle structure centralisée érode le principe d'autonomie des agents, dans la mesure où la matrice semble « dicter » à l'agent ce qu'il doit faire et avec qui sans lui laisser apparemment beaucoup d'alternatives. Or, en pratique, il faut tempérer cette observation :

— En premier lieu, nous travaillons en simulation, ce qui implique que le cadre comportemental donné aux agents est fixé (par le modèle thématique) : à ce titre, il est essentiel, pour des raisons de lisibilité, de disposer dans le modèle conceptuel d'une vue globale de tous les comportements qui peuvent avoir lieu ; quant au modèle computationnel, il est plus efficace de centraliser l'ensemble des affectations d'interactions au niveau par exemple de l'environnement que de les répartir au sein de chaque agent. Néanmoins, si l'on souhaitait utiliser l'approche orientée interactions dans le cadre d'agents physiquement distribués (par exemple des robots), rien ne s'oppose à doter chaque agent de la ligne et de la colonne de la matrice qui le concernent.
— Deuxièmement, la matrice ne fait que décrire des potentialités : pour qu'une interaction se réalise effectivement, il faut entre autres vérifier des gardes de distance et surtout les déclencheur et conditions des interactions : or ces dernières sont exprimées, comme nous l'avons vu, au moyen de primitives, ce qui garantit l'autonomie de chaque agent à ce stade.

Changements d'état : la matrice de mise à jour

En complément de la matrice d'interaction, nous avons été amenés à définir une **matrice de mise à jour**. Elle consiste en fait en un vecteur-colonne qui définit pour chaque famille d'agent la liste (ordonnée) des interactions à réaliser avant de commencer la sélection d'interaction proprement dite (voir le fonctionnement du moteur IODA, §2.2.3). Ces interactions sont toutes réflexives, et elles sont toutes effectuées (dans l'ordre donnée par leur priorité) si leurs déclencheur et condition sont vérifiés.

Cette matrice permet de définir les *changements d'état* qui affectent les agents, indépendamment du fait qu'ils puissent avoir une activité ou non dans la simulation. Ainsi, par exemple, si l'on souhaite introduire une notion d'âge chez les animaux de notre écosystème-jouet, il ne suffit pas d'introduire une interaction réflexive Vieillir, car elle serait en concurrence avec d'autres : un Loup ne pourrait vieillir que s'il n'effectue pas Manger par exemple, alors qu'il va de soi que le vieillissement se produit inéluctablement et quelle que soit l'activité du loup.

Pour cet exemple, on aurait donc la matrice de mise à jour suivante (tab. 2.2)

	UPDATE
Arbre	
Herbe	
Mouton	Vieillir (2) TomberMalade (1)
Chèvre	Vieillir (2) TomberMalade (1)
Loup	Vieillir (1)

TABLE 2.2 – La matrice de mise à jour qui complète la matrice d'interaction précédente (tab. 2.1). Ici on a fait en sorte que tous les agents vieillissent, et en outre les herbivores peuvent tomber malades (avec une certaine probabilité testée dans le déclencheur de l'interaction). Chaque interaction est associée à une priorité qui indique dans quel ordre (relatif) les interactions de mises à jour sont effectuées.

Si l'on souhaitait par exemple modifier le modèle conceptuel pour faire pousser les arbres et l'herbe, il suffirait de rajouter une interaction Pousser dans la matrice de mise à jour, sur la ligne de chacune de ces familles.

L'intérêt du tout-agent

La définition que nous avions donnée de l'agent (déf. II p. 5) était assez peu restrictive. Ce choix bien sûr n'était pas tout à fait innocent et ouvrait la voie à l'approche « tout agent » qui est au cœur de IODA.

Mais il s'appuie aussi sur des justifications plus profondes. La distinction que fait Ferber [110] entre les « vrais » agents, dotés de buts que les comportements visent à satisfaire, et les simples objets, utilisés comme ressources, fait en vérité peser de lourdes contraintes sur la révisabilité des modèles. Il en va de même pour des approches qui tentent d'introduire un soupçon d'activité dans les objets, par exemple *Agents & Artifacts* de Omicini *et al.* [168] ou d'autres recensées par Kubera [134] : elles introduisent un *typage rigide* des entités impliquées dans la simulation, de sorte que s'il devient nécessaire de doter un « objet » de comportements, une partie substantielle du code correspondant doit être réécrite.

Au contraire, dans l'approche IODA, toutes les entités sont traitées *de façon homogène*. En revanche, leur activité peut être caractérisée dynamiquement en fonction du contenu des matrices, et cela est fort utile pour écrire un moteur de simulation efficace (cf. §2.2.3). On peut ainsi attribuer trois propriétés (non mutuellement exclusives) aux agents IODA :

les agents actifs sont ceux qui ont une *ligne non vide* dans la matrice d'interaction, autrement dit ceux qui sont capables *d'effectuer des interactions* ;

les agents passifs sont ceux qui ont une *colonne non vide* dans la matrice d'interaction, autrement dit ceux qui sont susceptibles de *subir des interactions* ;

les agents labiles sont ceux qui ont une *cellule non vide* dans la matrice de *mise à jour*, autrement dit ceux qui peuvent changer d'état en dehors de tout autre comportement.

Dans notre exemple (tab. 2.1 et 2.2), tous les agents sauf l'Arbre sont actifs ; tous sont passifs car ils peuvent subir des interactions, et tous sauf l'Herbe et l'Arbre sont labiles car ils peuvent changer d'état (Vieillir).

Comme ces caractéristiques sont déterminées dynamiquement par le moteur, changer l'affectation des interactions aux familles d'agents (donc transformer des « objets » en « agents » ou vice-versa, selon la terminologie classique) n'a aucun coût pour le concepteur du modèle (sous réserve que les primitives correspondantes aient été écrites) : la simulation peut être relancée aussitôt.

Généricité et diversité des comportements : les primitives

Exprimer les comportements indépendamment des agents, au moyen d'interactions, c'est évidemment permettre la *réutilisation* de ces interactions dans des contextes différents. Des bibliothèques d'agents et d'interactions peuvent être développées séparément et mise en relation seulement lors de la construction du modèle.

Néanmoins, cela ne doit pas uniformiser les *comportements* manifestés par les agents dans les simulations. Un loup ne mange pas comme le fait un mouton. Ce respect de la diversité des comportements est assuré par le mécanisme des **primitives**.

Définition VI — Primitive.
Une primitive est une fonction ou une procédure qui est *abstraite* dans la définition de l'interaction, et *implémentée concrètement* dans le code de l'agent (ou de l'environnement) [2].

Par exemple, dans l'interaction Manger donnée en exemple ci-dessus (cf. fig. 2.2), la primitive `A_FAIM` doit être implémentée par tout agent qui peut être source de l'interaction. Mais elle peut être codée d'une façon arbitrairement complexe, par exemple comme la comparaison d'une variable à un seuil chez les herbivores, ou comme la durée écoulée depuis le dernier repas chez le loup, ou encore comme une fonction de l'heure chez l'homme, etc.

Cette stratégie de délégation de la variabilité comportementale aux primitives des agents reflète simplement le fait que les *modalités* selon lesquelles un agent évalue une situation ou réalise une action dépendent bien évidemment de ses spécificités perceptives, motrices et cognitives. En outre, d'un point de vue logiciel, il s'agit simplement d'une adaptation du concept de *polymorphisme* des langages orientés objets. Nous verrons dans les exemples applicatifs (chap. 3) l'intérêt de cette méthode.

Relations entre familles d'agents

Dans de nombreuses disciplines, certaines entités manipulées sont liées entre elles par des relations d'espèce à sous-espèce (qui peuvent prendre des formes variées : catégories et sous-catégories socio-professionnelles, familles et sous-familles de molécules, genres, espèces, variétés d'êtres vivants, etc.). D'un point de vue sémantique, il s'agit de la classique **relation sorte-de** (*kind-of*). Dans la mesure où cette relation se rencontre fréquemment dans les modèles thématiques, il est nécessaire d'offrir la possibilité de la représenter aisément dans le modèle conceptuel, et si possible de la conserver dans le modèle computationnel.

Dans IODA, la relation *sorte-de* peut être spécifiée dans l'écriture de la matrice d'interaction, simplement en l'indiquant par le symbole « : ».

La relation *sorte-de* ainsi construite est indépendante du langage utilisé pour implémenter la simulation, elle permet donc à une famille d'agents d'être une sous-famille de plusieurs autres groupes. Dans sa thèse, Kubera [134, chap. 8] a proposé des opérateurs permettant l'ajout, la modification et même la suppression d'interactions et décrit les algorithmes utilisés pour ce faire. Il est ainsi possible d'exprimer de façon très synthétique la façon dont familles et sous-familles diffèrent, y compris en introduisant des exceptions (comportements présents dans la famille mais pas dans une sous-famille).

Sources \ Cibles	∅	Arbre	Herbe	Mouton	Chèvre	Loup
Arbre Herbe	Pousser (0) SePropager (1)					
Animal	Mourir (4) Errer (0)					
Herbivore : Animal			Manger (2 ; 0)			
Mouton : Herbivore				Procréer (1 ; 0)		
Chèvre : Herbivore		Grimper (3 ; 0)			Procréer (1 ; 0)	
Loup : Animal				Manger (3 ; 1)	Manger (2 ; 1)	Procréer (1 ; 0)

	UPDATE
Arbre	
Herbe	
Animal	(Vieillir ; 2)
Herbivore : Animal	(TomberMalade ; 1)
Mouton : Herbivore	
Chèvre : Herbivore	
Loup : Animal	

TABLE 2.3 – Réécriture des matrices d'interaction et de mise à jour précédentes (tab. 2.1 et 2.2) avec l'utilisation de relations *sorte-de*. Ici le Mouton est une sorte d'Herbivore donc il bénéficie automatiquement de la capacité à effectuer Manger sur les agents Herbe ; les Herbivores sont eux-mêmes une sorte d'Animal, etc.

2.2.2 Un cadre méthodologique

Le projet IODA comporte également un volet méthodologique, qui a fait l'objet de nombreux développements par KUBERA [134]. De façon succincte, on peut décrire la méthode de conception de simulation liée à IODA comme une approche descendante « par paliers », qui ne contraint pas l'utilisateur à suivre des étapes dans un ordre rigide, mais lui permet au contraire de procéder par approfondissements successifs de parties du modèle (fig. 2.4 p. 28).

On doit dans un premier temps définir la topologie de l'environnement, le nom des familles d'agents et des interactions, ce qui permet d'écrire d'une part les matrices d'interaction et de mise à jour, et d'autre part les interactions elles-mêmes. L'affectation d'interactions aux agents avec une priorité et une garde de distance, ainsi que les noms de primitives utilisées pour ces interactions permettent alors de déterminer quelles primitives doivent être implémentées par chacune des familles d'agents et par l'environnement. Ces primitives à leur tour peuvent s'appuyer sur des caractéristiques propres aux agents ou à l'environnement et ainsi suggérer les structures de données nécessaires.

2.2.3 Un cadre algorithmique : le moteur IODA

Le moteur de simulation a pour fonction le contrôle des comportements des agents et de la dynamique de l'environnement ; il existe de nombreuses façons de le réaliser. En tout état de cause, un simulateur doit pouvoir gérer la succession des actions

dans le temps et la concurrence de l'activité des agents. Le formalisme DEVS (*Discrete Event Systems Specification*) [219] par exemple repose sur l'ordonnancement des divers événements qui peuvent survenir. En simulation multi-agents, on s'appuie en général plutôt sur des approches à **temps discret**, dans lesquelles le temps est divisé en intervalles de durée constante δt appelés **pas de temps**. Le temps de la simulation est donc représenté par un intervalle entier $[0, T_{\max}]$; au pas de temps t_i, le temps écoulé depuis le début de la simulation est donc $t_i \delta t$. Si cette représentation du temps est fort différente du temps continu de la physique, c'est en revanche celle que nous savons le plus facilement construire dans les systèmes artificiels, de la clepsydre à l'horloge atomique.

Reste à gérer la simultanéité des actions des agents au sein de chaque pas de temps. Lorsque l'on construit un outil de simulation, il est indispensable que les expériences réalisées soient *reproductibles*. En effet, le simulateur doit être considéré avec la même rigueur qu'un appareil de mesure en physique, dont il est essentiel de connaître *l'incertitude*. Si les résultats d'une expérience varient, il faut pouvoir faire la part entre les variations intrinsèques au phénomène étudié, celles qui sont introduites à dessein par exemple sous forme de tirages pseudo-aléatoires, et enfin les *biais* qui résultent d'une maîtrise incomplète de l'outil utilisé [12]. Entre autres biais qu'il est facile d'éliminer, on trouve toute forme de délégation de l'ordonnancement des actions au langage utilisé ou au système de la machine (par exemple, l'usage de *threads* en Java).

Pour construire un tel ordonnanceur, il ne reste que deux méthodes :

— **L'approche synchrone** consiste à évaluer toutes les actions que souhaitent réaliser les agents à partir d'une perception identique de l'environnement et de l'état des autres agents, puis à calculer la résultante de toutes ces actions pour définir l'état du monde au pas de temps suivant (comme dans le jeu de la vie [116]). Le représentant le plus abouti de cette approche pour la simulation multi-agents est le modèle IRM4S de Michel [154] qui découpe le pas de temps de simulation en une phase d'influences émises par les agents et une phase de réaction pour en calculer la synthèse. Toutefois, la difficulté majeure reste la définition de cette fonction de réaction.

— **L'approche séquentielle** consiste à faire percevoir et agir chacun des agents l'un après l'autre au cours du même pas de temps (méthode qu'on appelle également « tour de parole »). Ce procédé, qui ne cherche pas à garantir que les agents perçoivent le « même » monde durant un pas de temps et par conséquent ne vise pas à offrir une véritable simultanéité, est en revanche plus simple à mettre en œuvre que l'approche synchrone et se rapproche statistiquement d'une méthode événementielle, pour peu que l'on réordonne aléatoirement les agents au début de chaque pas de temps. C'est cette approche que nous avons retenue pour le moteur IODA.

Les connaissances représentées par des interactions et leur affectation aux familles d'agents sous forme matricielle, font l'objet d'un traitement générique par un moteur de simulation. Le moteur le plus simple qui puisse être construit sur cette base consiste à évaluer, pour chaque agent, quelles sont les interactions réalisables en fonction des agents perçus dans son voisinage, puis à choisir l'une de ces interactions possédant le plus haut niveau de priorité et à l'effectuer sur une des cibles possibles.

La mise en œuvre concrète n'est pas si facile. Si l'on souhaite éviter l'introduction de biais dans les résultats de simulation, il est préférable que le moteur prenne en charge explicitement un certain nombre des choix d'implémentation qui pourraient être faits implicitements par le programmeur [12] : le moteur doit donc être *paramétrable* et proposer des *politiques* (ou des stratégies) chaque fois qu'un choix doit être fait. Par exemple, il est nécessaire de prévoir à chaque pas de temps un *tour de parole* pour proposer à chaque agent d'agir : l'ordonnancement choisi pour ces agents constitue une politique, et la meilleur politique à suggérer par défaut est celle qui conduit au maximum d'équité, c'est-à-dire un réordonnancement aléatoire des agents à chaque pas de temps. De même, lorsqu'un agent recense les couples `(interaction, cible)` pour lesquels il peut agir comme source, il lui faut ensuite choisir une interaction et une cible : là encore, une politique doit être définie.

En outre, le choix du *tout agent*, s'il s'avère très fécond pour la conception des modèles, risquait de grever lourdement les performances du moteur. Nous avons donc élaboré un algorithme de simulation qui permet de tenir compte dynamiquement du caractère actif, passif ou labile des agents présents dans l'environnement [39, 2]. Le principe général consiste, chaque fois qu'un agent est introduit dans l'environnement où en est retiré, ou chaque fois qu'a lieu un changement des matrices d'interaction ou de mise à jour, à réactualiser des listes recensant les agents actifs, passifs et labiles. Il suffit alors d'utiliser les listes appropriées pour chacune des opérations qui interviennent dans la simulation d'un pas de temps, comme suit :

1. Phase de mise à jour des agents
   Pour tous les agents *labiles* a_ℓ (ordonnés selon une politique Π_ℓ) :
   — Pour toutes les interactions I_k de la matrice de mise à jour de a_ℓ, triées par priorité décroissante :
      — si les déclencheur et condition de I_k sont vérifiés pour a_ℓ, alors effectuer les actions de I_k
2. Phase de sélection d'interaction
   Pour tous les agents *actifs* a_{act} (ordonnés selon une politique Π_{act}) :
   (a) a_{act} perçoit dans son voisinage des agents *passifs* $V = \{a_i\}$;
   (b) on retire de V ceux qui ne peuvent pas être cible de a_{act} ;
   (c) pour toutes les interactions $\{I_1^p, ..., I_n^p\}$ de la matrice d'interaction de a_{act}, du même niveau de priorité p (en commençant avec la plus forte valeur de p) :
      i. on calcule les couples d'interactions réalisables $\mathcal{R}^p = \{(I_k^p, a_i)\}$ pour lesquels les déclencheur et condition de I_k^p sont vérifiés pour a_{act} et a_i (en prenant $a_i = a_{\text{act}}$ si I_k^p est réflexive) ;
      ii. si $\mathcal{R}^p \neq \emptyset$, on applique une **politique de sélection de cible** $\Pi_{\text{TSP}} : \mathcal{R}^p \mapsto (I^\star, L^\star)$ qui choisit une interaction à réaliser et une liste $L^\star$ d'agents cibles (vide si $I^\star$ est réflexive), puis on exécute les actions de $I^\star$ sur $L^\star$;
      iii. sinon on recommence avec les interactions de priorité inférieure à p.

Ce moteur s'est avéré parfaitement capable de réaliser des simulations performantes tout en assurant une réduction du risque de biais au moyen de diverses politiques [12, 2].

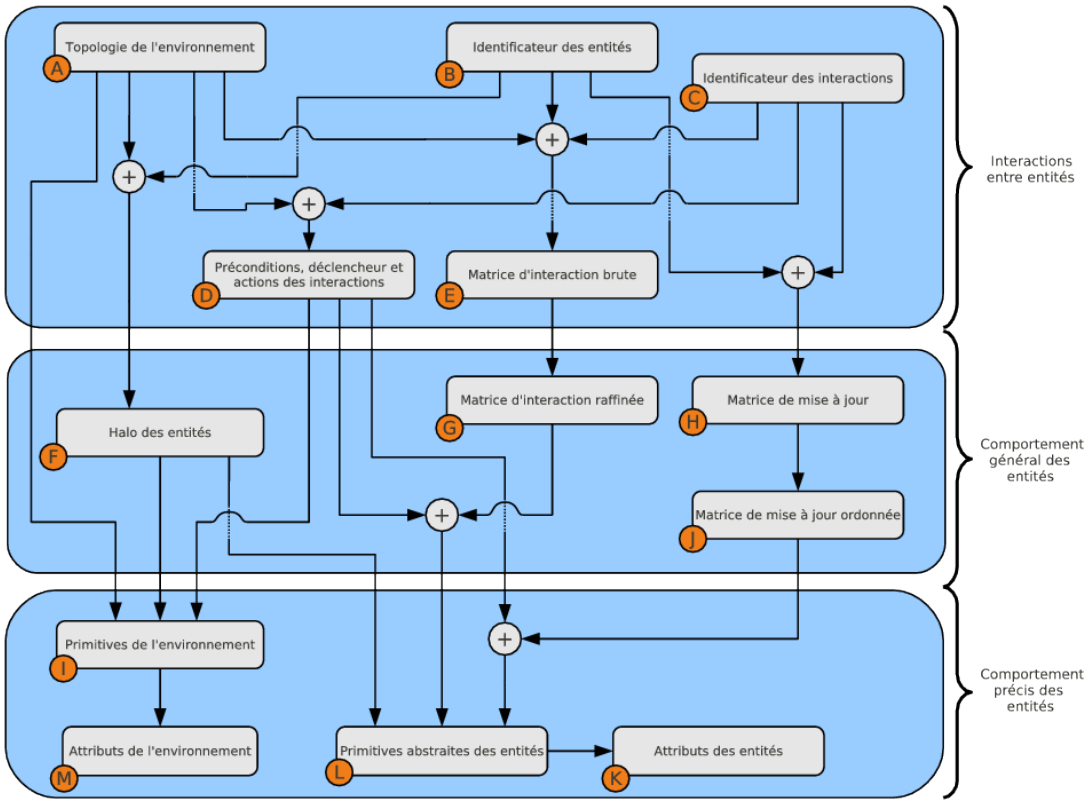


FIGURE 2.4 – Étapes de conception et d'implémentation d'un modèle avec IODA. Une assez grande souplesse est laissée au concepteur dans une approche descendante qui raffine et détaille pas à pas les divers points clefs (emprunté à KUBERA [134]).

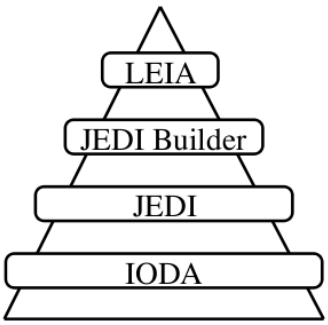


FIGURE 2.5 – La pyramide des outils IODA : la plateforme JEDI, l'environnement de développement et de génération de code JEDI-Builder, et LEIA, le navigateur dans l'espace des simulations.

2.3 L'univers IODA : quelques outils

Une méthode, un cadre formel, des algorithmes ne valent, nous l'avons dit, que s'ils sont implémentés, testés, appliqués à des situations réelles. Autour de IODA, ou plutôt au-dessus, se sont donc construits un certain nombre d'outils, le premier étant une plateforme d'expérimentation, JEDI. La séparation déclaratif/procédural permet de générer automatiquement une grande partie du code de simulation à partir du modèle conceptuel : c'est le rôle de JEDI-Builder. Enfin, pour un ensemble d'agents et d'interactions donné, il est possible de générer automatiquement des matrices d'interaction et de soumettre les simulations qu'elles engendrent à un processus évolutionniste : c'est ce que fait LEIA, un navigateur dans l'espace des simulations. La figure 2.5 résume cette pyramide d'outils dont nous allons dire quelques mots.

2.3.1 JEDI : une plateforme d'expérimentation

☞ `http://www.lifl.fr/SMAC/projects/ioda/jedi/`

La méthode IODA se veut particulièrement adaptable aux besoins de ses utilisateurs dans divers domaines applicatifs, tout particulièrement lorsqu'ils ne sont pas formés à l'algorithmique ou à la programmation. Il serait paradoxal, dès lors, de vouloir diffuser cette approche par le biais d'une plateforme unique, qui plus est écrite en Java et d'un haut niveau de technicité. De plus, le meilleur moyen de s'assurer que l'approche orientée interactions est indépendante d'un langage et même du paradigme objet, consiste à réaliser des implémentations du moteur sous des architectures diversifiées. Nous en verrons d'ailleurs quelques exemples plus loin (cf. §3.2.1, §3.3.1 et §3.3.2).

Le rôle de la plateforme JEDI (pour : *Java Environment for the Design of agent Interactions*), entièrement développée par Kubera [134], n'a donc pas pour objectif d'être LA plateforme universelle de diffusion de l'approche. En revanche, elle est destinée à expérimenter tous les détails de l'approche orientée interactions, tous ses algorithmes, toutes ses politiques, et à en donner une implémentation aussi fiable et efficace que possible. C'est en quelque sorte pour IODA une « théorie matérialisée » selon le mot de Bachelard [59, p. 16]. L'étude des biais de simulation et des façons de les éviter [12] a été au cœur de la conception de cette plateforme. Actuellement JEDI compte près de 15 000 lignes de code réparties dans une centaine de classes et d'interfaces, et peut faire tourner plusieurs centaines de milliers d'agents. Elle est extrêmement modulaire et actuellement distribuée par l'équipe SMAC sous la forme d'une archive JAR. Nous donnons sur la figure 2.6 un diagramme de classes *extrêmement simplifié* du « cœur » de JEDI.

Au-delà de ces considérations de génie logiciel, le point le plus important résulte de la séparation déclaratif/procédural : il est en effet possible d'automatiser dans une très large mesure le passage d'un modèle conceptuel IODA au modèle computationnel correspondant et à son implémentation au sein de JEDI, grâce à un environnement de développement intégré qui s'appuie sur la plateforme : JEDI-Builder (fig 2.7).

JEDI-Builder met en œuvre la méthodologie associée à IODA (cf. §2.2.2) en permettant à l'utilisateur de concevoir son modèle de simulation de façon interactive,

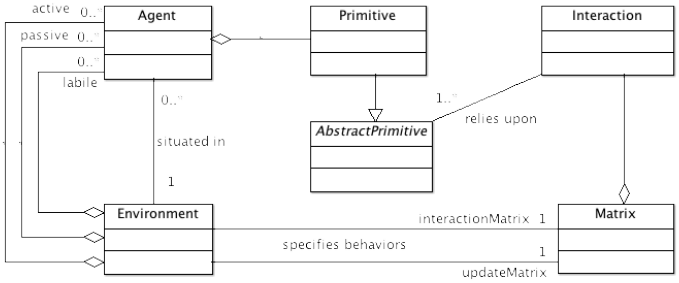


FIGURE 2.6 – Diagramme de classes simplifié à l'extrême du cœur du simulateur JEDI.

notamment en créant la matrice d'interaction par *glisser-déposer*, puis en analysant les primitives nécessaires à chaque classe d'agent et en générant soit le code complet, soit le squelette du code à écrire. La figure 2.8 montre que les tâches dévolues au seul programmeur sont réduites d'une part à la définition des déclencheur, condition et actions des interactions, et d'autre part à l'écriture du code des primitives : la tâche, lourde, de concevoir une simulation multi-agents, est donc en partie automatisée, le reste étant découpé en petites portions de code indépendantes les unes des autres.

Enfin, JEDI contribue à la diffusion de l'approche orientée interactions, sous la forme d'une *applet* qui met en scène d'une part des agents identifiés par leur couleur et placés dans une grille 2D, d'autre part des interactions très simples (comme se dupliquer, tuer un autre agent, etc.). Cet outil à vocation pédagogique permet de construire des simulations par des modifications incrémentales de la matrice d'interaction pour en voir aussitôt le résultat (fig. 2.9), et en dépit de la simplicité de ses comportements, autorise la reproduction de phénomènes assez intéressants comme des dynamiques de ségrégation à la SCHELLING [193] ou de réactions chimiques oscillantes du genre BZ [64, 220].

☞ `http://www.lifl.fr/SMAC/projects/ioda/jedi/demonstration.php`

2.3.2 LEIA : un navigateur dans l'espace des simulations

Dans une simulation multi-agents, **l'espace des simulations possibles** est généralement réduit à l'espace des *paramètres* qui peuvent modifier l'état des agents ou le déroulement des comportements [204]. Il en va tout autrement avec IODA, puisque l'affectation des interactions aux familles d'agents fait partie des « éléments » constitutifs du modèle. L'espace des simulations possibles s'accroît donc considérablement.

Loin de constituer un handicap pour la conception de simulations, c'est au contraire une situation particulièrement favorable. En effet, dans de nombreux domaines, il peut être intéressant d'examiner des *variantes* d'un modèle donné pour savoir si les hypothèses qui ont été formulées et testées sont nécessaires, suffisantes, ou s'il existe des alternatives radicalement différentes pour rendre compte du même phénomène.

L'outil LEIA (pour : *LEIA lets you Explore your Interactions for your Agents*) conçu par GAILLARD *et al.* [4] est une sorte de « navigateur » qui permet d'explorer l'espace des

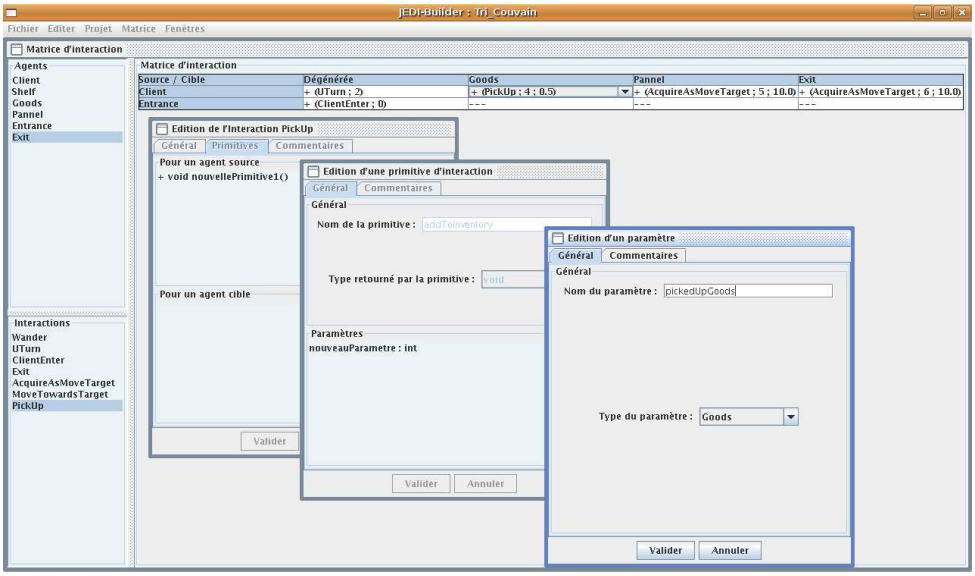


FIGURE 2.7 – Vue de l'IDE JEDI-Builder (emprunté à KUBERA [134]).

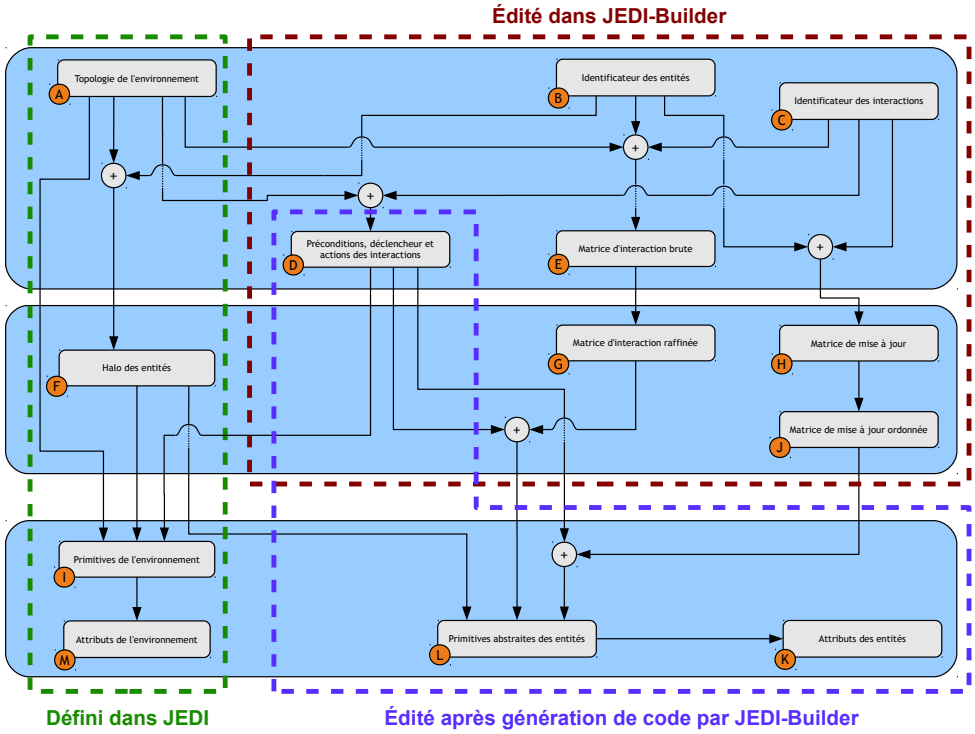


FIGURE 2.8 – Répartition des tâches entre JEDI, JEDI-Builder et le développeur lors de l'utilisation de JEDI-Builder pour l'implémentation d'un modèle IODA (emprunté à KUBERA [134]).

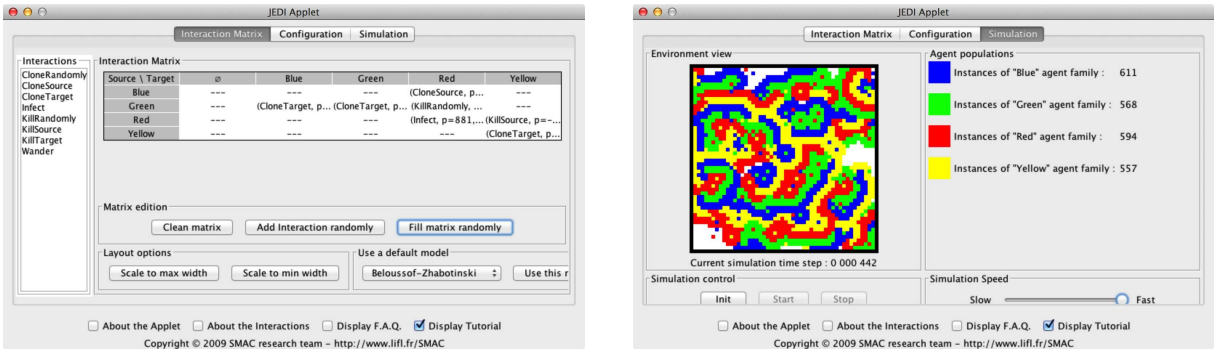


FIGURE 2.9 – L'applet de démonstration de JEDI, dans laquelle on peut interactivement construire une matrice d'interaction à partir des agents et des interactions existantes (à gauche) et en visualiser immédiatement le résultat dans la simulation (à droite).

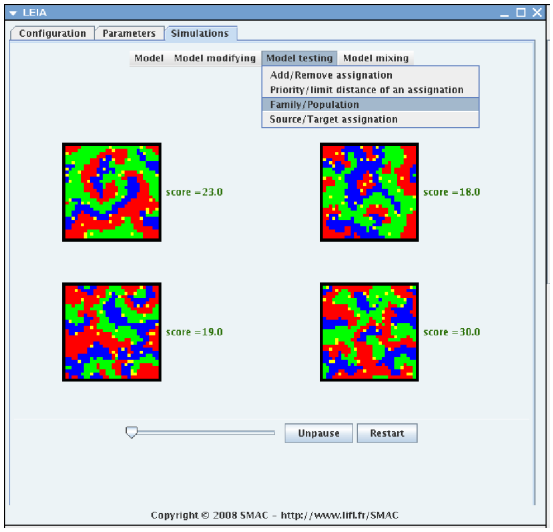


FIGURE 2.10 – Une vue du navigateur LEIA, avec ici quatre simulations lancées en parallèle. L'utilisateur peut choisir un mode de sélection automatique ou interactif, et définir ses propres critères d'évaluation en plus ou à la place des critères intrinsèques prédéfinis.

simulations de façon soit automatique, soit interactive. Il consiste à définir, pour un ensemble d'agents et un ensemble d'interactions fixés, un modèle dit « de base ». Ce modèle est transformé par un certain nombre d'opérateurs (qui peuvent être choisis selon les objectifs visés), tels que le déplacement d'une interaction d'une case de la matrice vers une autre, l'ajout ou la suppression d'une interaction dans une case, la modification de la priorité, etc., de façon à produire N modèles « dérivés ». Ces N modèles sont exécutés dans autant d'instances de JEDI en parallèle, et un *score* leur est attribué. Ce score peut être calculé selon des critères intrinsèques au déroulement de la simulation (existence de cycles dans les comportements, densité des agents, évolution des populations, etc.) ou selon des critères propres au domaine considéré. Enfin, les meilleurs modèles (sélectionnés sur leur score ou par un observateur humain) sont hybridés à l'aide d'opérateurs de fusion de matrices afin de créer un nouveau « modèle de base », et le processus est itéré autant que nécessaire (fig. 2.10).

Un tel outil constitue en pratique une première ébauche de générateur de SMA, capable à partir d'une bibliothèque d'agents et d'une bibliothèque d'interactions de produire un nombre colossal de simulations qui diffèrent considérablement les unes des autres et peuvent satisfaire des contraintes portant aussi bien sur la structure du modèle que sur les comportements observés. On voit par là que la séparation, dans le modèle, des aspects déclaratifs et procéduraux, ainsi que des diverses connaissances (agents et interactions) permet une manipulation aisée du niveau « méta » au sens de Pitrat [174].

Enfin, avouons aussi que LEIA peut avoir un usage *créatif* : c'est un outil d'exploration libre d'un champ de possibilités extrêmement vastes, et à ce titre il permet de découvrir des associations entre interactions et agents dont le résultat peut être surprenant. Il peut donc être vu comme un support d'expériences créatives comme le suggère Hofstadter [131].

2.4 Bilan

L'approche orientée interactions, en procédant essentiellement à une séparation des aspects déclaratifs et procéduraux d'un modèle de simulation, et en s'appuyant sur le polymorphisme des primitives de perception et d'action, permet d'améliorer la qualité des simulations multi-agents, suivant les critères que nous avions définis précédemment :

— **Fiabilité :** les choix de conception et d'implémentation sont rendus aussi explicites que possible, quitte à ce que des choix par défaut soient proposés.
— **Efficacité :** des algorithmes appropriés permettent de garder de bonnes performances en temps d'exécution, quelles que soient les modifications apportées dynamiquement au modèle.
— **Manipulabilité :** grâce à l'automatisation du passage du modèle conceptuel au code, ce qui reste à programmer se réduit à des capacités élémentaires de perception et d'action.
— **Intelligibilité** les comportements effectués par les agents sont lisibles d'un coup d'œil sur les matrices d'interaction et de mise à jour ; les interactions sont exprimées en clair, sous formes de règles conditions/actions.
— **Modularité** : elle découle de la séparation agents/interactions.

Il nous reste à ce stade un point à démontrer : la **polyvalence** de notre approche, autrement dit sa capacité à permettre aussi bien la conception de simulations à visée scientifique que la création de mondes artificiels. C'est l'objet du prochain chapitre, où nous verrons la méthode IODA appliquée à des situations fort diverses.

Chapitre 3

La simulation orientée interactions : applications

Concevoir un cadre original pour la simulation de systèmes complexes et large échelle nous a demandé, comme on vient de le voir, une élaboration théorique et pratique assez importante, dont les développements (JEDI-Builder, LEIA) montrent pleinement l'intérêt d'un point de vue informatique. Néanmoins, nous nous sommes attachés, tout au long de ces travaux, à illustrer également l'intérêt de notre approche pour les disciplines visées par la simulation multi-agents.

Nous allons en particulier détailler deux projets très différents dans lesquels l'approche IODA a été particulièrement utile : le premier a consisté à tester des hypothèses en biologie cellulaire, le second à réaliser un *Serious Game* immersif destiné à la formation d'étudiants en marketing. Enfin, pour clore ce chapitre nous présenterons les efforts déployés pour diffuser les concepts, algorithmes et méthodes de l'approche orientée interactions, notamment à travers un outil de démonstration pédagogique (le projet Galaxian) et le développement d'une extension IODA pour la plateforme de simulation NetLogo.

3.1 Formulation d'hypothèses en biologie

En premier lieu, nous allons dans cette partie résumer l'activité menée en biologie et montrer comment la méthode IODA a montré aussi bien ses capacités à investir ce domaine qu'à y apporter une aide à la formulation d'hypothèses.

3.1.1 Contexte et objectifs

Les travaux que nous présentons ici s'inscrivent dans des recherches menées en 2006–2007 au sein d'un groupe de travail pluridisciplinaire, en vue de proposer et comparer divers modèles pour rendre compte du fonctionnement de l'horloge circadienne d'une algue verte : *Ostreococcus tauri*, étudiée par l'équipe « Rythmes Circadiens et Division Cellulaire » de François-Yves Bouget (Observatoire Océanologique de Banyuls-sur-Mer). Ce groupe était composé de diverses équipes de l'Université Lille 1 :

— le laboratoire de Physique des Lasers, Atomes, Molécules ;
— l'Unité de Glycobiologie Structurale et Fonctionnelle ;
— le Laboratoire d'Informatique Fondamentale de Lille (équipes Calcul Formel et SMAC).

Les compétences couvertes vont ainsi de la dynamique non linéaire à la simulation multi-agents, en passant par le calcul formel pour la réduction des modèles équationnels.

La plupart des êtres vivants voient leur activité régulée par divers rythmes, certains appelés *circadiens* étant caractérisés par une période proche de 24h. Ces rythmes, capables de se synchroniser avec des variations environnementales (alternance jour/nuit principalement), sont *endogènes*, c'est-à-dire produits par des processus biologiques internes : en l'occurrence, ils résultent souvent de mécanismes de régulation de l'expression de certains gènes [126]. Rappelons que les gènes (codés par l'ADN) font l'objet d'une **transcription** en une séquence d'ARNm (ARN messager) qui à son tour passe par une phase de **traduction** en protéines, effectuée par des structures cellulaires appelées *ribosomes*. La régulation de l'expression d'un gène peut intervenir à une ou plusieurs de ces étapes en activant ou en réprimant la transcription du gène, la traduction de l'ARNm, ou même l'activité de la protéine (voir par exemple les schémas ci-après, fig. 3.2 ou 3.3).

Le grand nombre d'espèces moléculaires susceptibles de jouer un rôle dans les mécanismes de l'horloge complique l'étude de ces réseaux de régulation. Toutefois, l'algue verte unicellulaire *Ostreococcus tauri* [85], qui est le plus petit eucaryote (cellule à noyau) connu, possède un génome très compact [89], donc relativement moins d'acteurs moléculaires.

Les travaux de simulation que le groupe de travail a menés visaient à :

— examiner, en termes de systèmes dynamiques, la *plausibilité des hypothèses biologiques* concernant la structure de l'horloge moléculaire ;
— *comparer plusieurs modèles* et identifier les acteurs moléculaires susceptibles d'intervenir dans l'horloge circadienne ;
— étudier diverses hypothèses relatives aux voies d'entrée des signaux lumineux ;
— suggérer aux biologistes, sur la base du comportement des systèmes simulés, des expériences complémentaires pour conforter ou infirmer leurs hypothèses.

La méthode prédominante pour la modélisation de réseaux de régulation génétique consiste à décrire l'évolution dans le temps des concentrations des diverses molécules impliquées, au moyen de systèmes d'équations différentielles ordinaires (ODE) [123, 145, 73]. Les ODE permettent une étude analytique, ainsi qu'une prédiction de résultats d'ensemble par intégration numérique. De plus ils fournissent un cadre uniforme de modélisation des circuits de régulation génétique [113]. En revanche, ils reposent sur un ensemble d'hypothèses qui ne sont pas toujours tenables :

— la notion de *concentration* est supposée toujours valide, alors que le nombre de molécules mis en jeu peut devenir très faible, ou que la croissance de la taille de la cellule durant sa réplication provoque en fait une dilution, etc. ;
— les équations sont déterministes : se pose donc la question de la robustesse de ces systèmes au bruit et aux événements à caractère stochastique (fixation d'un régulateur transcriptionnel sur l'ADN par exemple) ;

— la cellule est considérée comme *homogène*, il est donc difficile de modéliser les hétérogénéités spatiales, comme le simple fait que transcription et traduction ont lieu dans des compartiments cellulaires différents.

Pour compenser ces problèmes, de nombreux auteurs soumettent leurs modèles à des simulations stochastiques pour évaluer l'influence du bruit [124] ou introduisent des délais dans les équations pour remplacer le transport de molécules au sein de la cellule [143].

Or la simulation multi-agents, par sa nature intrinsèquement stochastique et spatialisée, est un outil adéquat pour la modélisation de ces phénomènes. En particulier, la méthode IODA se prête volontiers à la formulation d'hypothèses successives et la révision des modèles associés. Nous allons d'abord expliquer la méthode suivie pour la conception des modèles et leur validation, avant de résumer les deux modèles d'horloge retenus et les résulats de simulation obtenus ; de plus amples détails sont disponibles dans l'article [30].

3.1.2 Contraintes et méthodologie

De nombreuses contraintes pèsent sur la forme que peuvent prendre les modèles et sur la calibration des simulations.

Contraintes sur les données

En premier lieu, l'acquisition des données expérimentales relatives aux rythmes circadiens d'*O. tauri* est longue et coûteuse : tracer une courbe significative pour l'expression de l'ARNm demande au minimum un prélèvement, toutes les 3h pendant une semaine, d'une fraction de la culture de cellules, qui est alors broyée. L'ARNm est amplifié pour mesurer *son niveau relatif*. La mesure des protéines peut être optique (en ayant fusionné le gène cible avec un gène de molécule luminescente) ou effectuée par micropuce (avec un coût financier important). De plus les expériences sont limitées dans le temps car après une semaine de culture, les cellules sont trop nombreuses dans le milieu.

En aucun cas on ne dispose de mesure individuelle (au niveau d'une cellule) ni absolue : toutes les données concernent des populations de cellules. Outres les difficultés de mesures individuelles, les biologistes font en effet implicitement l'hypothèse que les millions ou milliards d'individus d'un milieu de culture sont interchangeables : issus d'une même souche, ils sont génétiquement identiques, et soumis aux mêmes conditions environnementales. Pourtant, lorsqu'une oscillation disparaît ou change de période et d'amplitude, on ne peut pas rigoureusement déterminer si c'est parce qu'il n'y a plus d'oscillations dans les cellules, ou parce que celles-ci ne font que se désynchroniser.

De plus les mesures ont une forte incertitude (fig. 3.1), on observe une importante variabilité quantitative d'une expérience à une autre, et les amplitudes sont normalisées par rapport à des gènes « de référence ». La comparaison entre l'expression de deux gènes est donc surtout qualitative, ce qui rend indétectables des oscillations de faible amplitude. Toutefois, les résultats qualitatifs expérimentaux sont reproductibles en termes d'acuité des pics, de déphasage entre courbes, etc. : c'est donc ces caractéristiques que l'on peut s'attacher à reproduire par simulation.

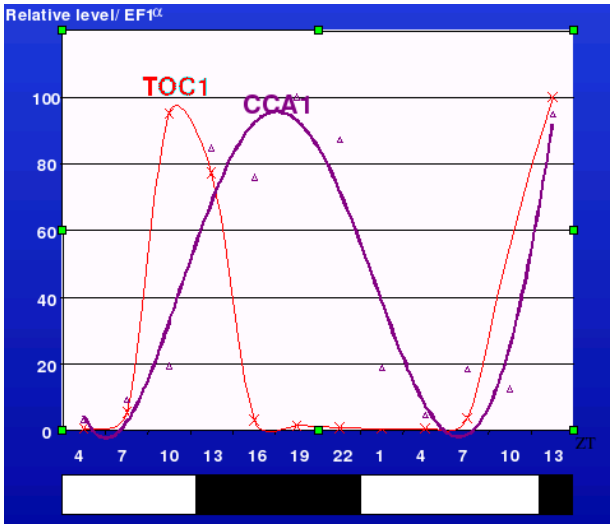


FIGURE 3.1 – Données expérimentales du groupe Horloge circadienne et cycle cellulaire de l'Observatoire océanologique de Banyuls, donnant les niveaux d'expression (quantité normalisée d'ARNm) de deux gènes d'*O. tauri* (TOC et CCA) en fonction du temps circadien (ZT), mesurés en alternance jour/nuit (bandes blanches et noires).

Contraintes sur les résultats

L'objectif des travaux de simulation menés soit par des méthodes équationnelles, soit par des méthodes multi-agents, était en premier lieu de déterminer des modèles minimaux d'horloge impliquant une régulation de la transcription génétique. Une question critique était celle du choix des *paramètres* de ces modèles (taux de transcription des gènes, demi-vie des ARNm et des protéines, etc.), étant donné que nous ne disposions pas de données biologiques précises. Deux voies étaient envisageables.

La première consiste à chercher à *identifier* ces paramètres par diverses méthodes d'apprentissage, en comparant les courbes simulées pour divers jeux de paramètres aux courbes expérimentales. Cela pourrait être envisagé dans les modèles équationnels, sans garantie toutefois d'obtenir des paramètres biologiquement plausibles, surtout à haut niveau de bruit et sans réduction préalable du modèle [72]. Dans le cas de simulations multi-agents, cette méthode n'a aucun sens puisque les échelles (d'espace, de nombre d'agents, de dimension, etc.) ne peuvent pas être respectées.

La démarche alternative consiste à trouver des jeux de paramètres, conformes aux ordres de grandeur connus dans la littérature, qui respectent les propriétés dynamiques des circuits de régulation génétique, en particulier pour les critères liés à l'évolution temporelle du système :

— le déphasage entre les courbes d'ARNm et de protéines doit être le même qu'observé expérimentalement ;
— le déphasage entre les courbes d'expression des acteurs du modèle à plusieurs gènes (déphasage entre l'ARNm de A et celui de B, entre la protéine A et la protéine B) ;
— la possibilité d'entraîner le système à une fréquence légèrement inférieure à sa fréquence naturelle (24h au lieu de 25h) ;

— la possibilité pour le système de continuer à osciller en lumière ou en obscurité continues.

Méthode de calibration des simulations

Pour calibrer les simulations, nous avons donc cherché à appliquer ces critères en procédant « à rebours » :

1. Déterminer des jeux de paramètres (biologiquement plausibles) donnant lieu à des oscillations ;
2. Calculer la période T^* (en nombre de pas de temps) de ces oscillations en régime libre : cette valeur est censée représenter la période naturelle de l'horloge d'*O. tauri* soit environ 25h ;
3. Introduire un signal lumineux de période T légèrement inférieure à T^* ; T représente par construction la durée d'une journée (24h) et permet ainsi d'établir la durée que représente un pas de temps ;
4. Vérifier que le système oscille en alternance jour/nuit, mais également en jour continu et en nuit continue, conformément aux observations biologiques ;
5. Vérifier le déphasage entre les courbes d'ARNm et de protéine d'un même produit (entre 2 et 5h) ;
6. Dans les modèles à plusieurs gènes, vérifier les déphasages entre les divers ARNm ou protéines ainsi que l'heure des pics pour les comparer aux données.

Un jeu de paramètres peut être éliminé dès qu'il ne satisfait plus à l'un de ces critères.

3.1.3 Recherche de modèles minimaux

Les modèles étudiés visaient à déterminer combien de gènes au minimum devaient être impliqués dans l'horloge. Nous avons donc d'abord examiné les caractéristiques d'une horloge construite à partir de la régulation d'un seul gène, puis de deux.

Modèle à 1 gène

La boucle de régulation la plus simple est composée d'un gène auto-réprimé (fig. 3.2) : les protéines issues de ce gène inhibent sa transcription. Il s'agit d'une rétroaction négative dont on peut s'attendre à ce qu'elle produise des oscillations : en effet, comme l'ARNm et les protéines de dégradent au cours du temps, le gène finit par repasser dans l'état libre et la transcription peut redémarrer.

Dans ce genre de modèle, on suppose ordinairement que le taux de dégradation de l'ARNm et des protéines est constant au cours du temps (d'où suit une loi exponentielle : $\frac{\mathrm{d}M}{\mathrm{d}t} = -\delta_M\ M$). Toutefois, l'étude menée dans notre groupe de travail par Boulier *et al.* [73] démontre analytiquement qu'il faut envisager des mécanismes supplémentaires pour obtenir un système oscillant : délais dans les équations [143], transport de protéines entre cytoplasme et noyau [142], modifications fonctionnelles des protéines [122] ou plus simplement dégradation des protéines par une

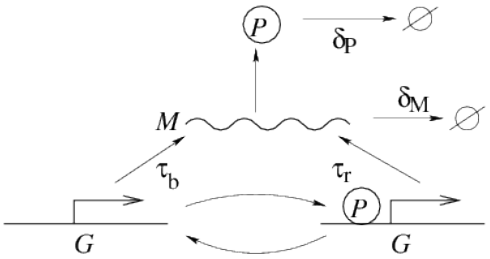


FIGURE 3.2 – Modèle le plus simple : gène réprimé par sa propre protéine. Le gène G à l'état non régulé (à gauche) est transcrit sous forme d'ARN messager (M) à un taux supposé constant au cours du temps (τ_b). M est à son tour traduit en protéine (P). M et P subissent une dégradation exponentielle au cours du temps (resp. δ_M et δ_P). La protéine P peut se fixer sur un site régulateur de G et réprimer sa transcription (à droite) : G est alors transcrit à un taux $\tau_r \ll \tau_b$ — cf. [30].

enzyme [123]. Faute d'indications biologiques spécifiques, c'est ce dernier processus, très bien connu, qui a été retenu.

Les simulations numériques de MORANT *et al.* [162] montrent d'ailleurs que des oscillations apparaissent facilement en introduisant un processus de dégradation enzymatique représenté par la réaction suivante : $P + E \rightleftharpoons C \rightarrow E$ dans lequel la protéine P forme avec une enzyme (protéase) E un complexe C plus ou moins stable, qui dégrade la protéine en libérant l'enzyme. En situation quasi-stationnaire, la dégradation des protéines suit alors la dynamique de MICHAELIS et MENTEN [153] : $\frac{\mathrm{d}P}{\mathrm{d}t} = -\frac{V_{\max} P}{K+P}$ (où $V_{\max}$ et K s'expriment en fonction des constantes cinétiques de la réaction). L'intérêt de ce mécanisme est de produire un effet de *saturation* lorsque la quantité de protéine à dégrader est très supérieure au seuil K, car on a alors $\frac{\mathrm{d}P}{\mathrm{d}t} \sim -V_{\max}$ (dégradation linéaire), alors que pour une quantité de protéines assez inférieures à K, on a au contraire $\frac{\mathrm{d}P}{\mathrm{d}t} \sim -\frac{V_{\max}}{K} P$ (dégradation exponentielle) : autrement dit, ce processus tend à instaurer deux régimes : un *niveau haut* de protéines caractérisé par une dégradation relativement faible et un *niveau bas* où la dégradation est plutôt forte.

Modèle à 2 gènes

Si le circuit à 1 gène auto-réprimé offre l'avantage de la parcimonie, il est biologiquement peu plausible car moins robuste : de fait plusieurs horloges circadiennes de plantes impliquent au moins deux gènes : ainsi, chez *Arabidopsis thaliana*, deux gènes centraux de l'horloge [190, 145] ont des homologues chez *O. tauri* : ce sont TOC et CCA, considérés comme de bons candidats par l'équipe de l'Observatoire océanologique. Le mécanisme présumé est le suivant : la protéine TOC active la transcription du gène CCA, tandis que la protéine CCA réprime la transcription de TOC (fig. 3.3 en posant A = TOC et B = CCA). Les résultats expérimentaux obtenus sur *O. tauri* montrent des oscillations sur ces gènes au niveau des transcrits et des protéines, aussi bien en alternance jour/nuit qu'en lumière ou en nuit continues (fig. 3.1).

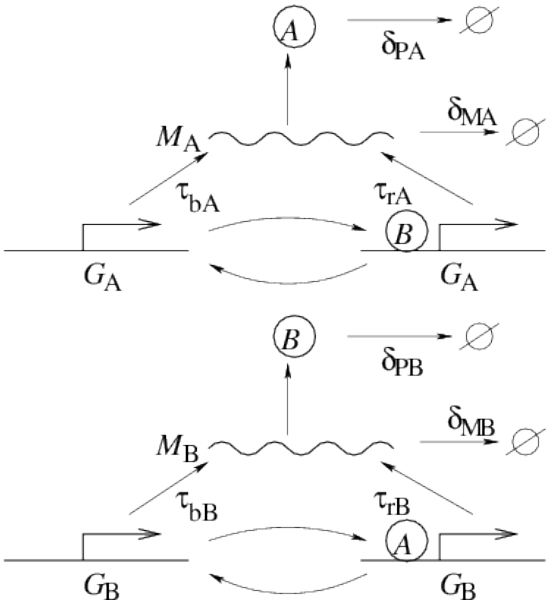


FIGURE 3.3 – Modèle à deux gènes, avec régulation croisée : les protéines A et B, produites respectivement par transcription et traduction des gènes G_A et G_B, régulent la transcription respective de B et A. Il peut s'agir d'une activation ou d'une répression.

Action de la lumière

Une horloge circadienne dispose d'un mécanisme de recalage sous l'action de la lumière. Le modèle d'horloge génétique *stricto sensu* s'accompagne donc d'hypothèses quant aux voies d'entrée de la lumière : sur quel(s) acteur(s) moléculaire(s) a-t-elle un effet (ARNm, protéine, métabolites intermédiaires...) et lequel (dégradation, modification de la fonction biologique, changement de compartiment cellulaire...) ? Toujours dans un souci de parcimonie, l'hypothèse de travail retenue en concertation avec les biologistes était délibérément simpliste : nous avons supposé que *la lumière agissait directement sur la protéine TOC en la dégradant*. Les éclairages apportés sur cette hypothèse par les résultats de simulation sont discutés au § 3.1.5.

3.1.4 Agentification *via* IODA et expériences

À partir de ces modèles thématiques à 1 ou 2 gènes, nous avons élaboré et simulé des modèles conceptuels multi-agents dans le cadre de la méthode IODA en évaluant la capacité des simulations à reproduire correctement la dynamique globale du système en fonction des acteurs et de leurs interactions, ainsi que ses réponses aux contraintes environnementales et aux perturbations endogènes.

L'environnement

L'environnement est l'intérieur d'une cellule unique, vue comme une surface torique 2D (pour s'affranchir des problèmes de frontière), avec une zone centrale qui joue le rôle de noyau et détermine la position des ribosomes.

Les agents

Les biologistes ont identifié les familles d'agents suivantes qui reflètent très directement les entités de leurs modèles :

G Les *gènes*, qui codent des protéines, peuvent être dans l'état libre, activé ou réprimé, chacun étant caractérisé par un *taux de transcription* qui donne la probabilité, à chaque pas de temps, d'être transcrit en ARNm.

M Les *ARNm*, issus de la transcription d'un gène, diffusent à l'intérieur de la cellule et subissent une décroissance exponentielle : autrement dit, à chaque pas de temps un agent ARNm a une probabilité constante δ_M de disparaître.

R Les *ribosomes*, placés à la périphérie du noyau, ont pour fonction d'effectuer la traduction d'un ARNm en la protéine correspondante.

P Les *protéines*, issues de la traduction des ARNm par les ribosomes, diffusent à l'intérieur de la cellule et peuvent soit subir une décroissance exponentielle (probabilité constante δ_P), soit être dégradées par des enzymes spécifiques appelées *protéases*. Une protéine peut par ailleurs jouer un rôle régulateur et activer ou inhiber la transcription d'un gène cible.

Δ Les *protéases* sont des enzymes qui diffusent dans la cellule et dégradent des protéines spécifiques (selon le processus de Michaelis et Menten [153]).

Bien évidemment, les nombres de molécules ne peuvent être strictement respectés, à la fois faute d'informations détaillées dans les mesures biologiques et pour des raisons évidentes de capacité de calcul. Nos agents représentent donc des sortes de *quanta* de molécules, un peu à la façon des agents « boule d'eau » utilisés en hydrologie par Servat [195].

Enfin, pour tenir compte du rôle de la lumière, deux solutions étaient envisageables : soit placer dans l'environnement une information globale « intensité lumineuse », soit *agentifier la lumière* au moyen d'agents « photons » (notés φ) : une fois encore la pertinence de ce choix sera discutée plus loin (§ 3.1.5).

Les interactions

Conjointement à la définition des agents, il a été assez facile aux biologistes de spécifier les différentes interactions susceptibles de se produire dans le système :

Mourir décroissance exponentielle : l'agent source peut disparaître spontanément avec une probabilité constante dans le temps ;

Diffuser diffusion de l'agent source dans la cellule (déplacement aléatoire) ;

Transcrire transcription d'un gène : le gène qui effectue cette interaction peut produire un ARNm avec une probabilité qui dépend de son état (libre, activé ou réprimé) ;

Détruire dégradation de l'agent cible par l'agent source ;

Traduire traduction par un ribosome d'un ARNm en la protéine correspondante ;

Reprimer l'agent source se fixe sur l'agent cible et change l'état de ce dernier en « réprimé » ;

Activer l'agent source se fixe sur l'agent cible et change l'état de ce dernier en « activé » ;

Avancer déplacement en ligne droite.

Formulation des hypothèses

La formulation des hypothèses concernant la structure de l'horloge s'exprime ensuite très simplement au moyen d'une matrice d'interaction : cette façon de représenter visuellement les interactions réalisables entre familles d'agents a été d'une grande aide pour expliciter l'expertise des biologistes. En particulier, on voit que le passage d'un modèle à l'autre consiste essentiellement à agrandir la matrice en y plaçant les interactions correspondant aux nouvelles hypothèses. Les tableaux 3.1 et 3.2 donnent les matrices d'interaction respectives des modèles à 1 gène auto-réprimé avec dégradation enzymatique de P et un modèle à 2 gènes avec décroissance exponentielle de TOC et CCA.

Sources \ Cibles	∅	G	M	P
G	Transcrire (0)			
M	Mourir (1) Diffuser (0)			
R			Traduire (0 ; 1)	
P	Diffuser (0)	Réprimer (1 ; 1)		
Δ	Diffuser (0)			Détruire (1 ; 1)
φ	Avancer (0)			Détruire (1 ; 1)

TABLE 3.1 – Matrice d'interaction d'un modèle à 1 gène G dont la protéine P réprime la transcription. Dans la situation présentée ici, les photons φ détruisent directement la protéine P. Les colonnes vides ne sont pas affichées.

Sources \ Cibles	∅	G_{TOC}	G_{CCA}	M_{TOC}	M_{CCA}	P_{TOC}
G_{TOC}	Transcrire (0)					
G_{CCA}	Transcrire (0)					
M_{TOC}	Mourir (1)					
M_{CCA}	Mourir (1) Diffuser (0)					
R				Traduire (0 ; 1)	Traduire (0 ; 1)	
P_{TOC}	Mourir (2) Diffuser (0)		Activer (1 ; 1)			
P_{CCA}	Mourir (2) Diffuser (0)	Réprimer (1 ; 1)				
φ	Avancer (0)					Détruire (1 ; 1)

TABLE 3.2 – Matrice d'interaction d'un modèle à deux gènes G_{TOC} et G_{CCA}, la protéine P_{TOC} activant la transcription du gène G_{CCA} et la protéine P_{CCA} réprimant la transcription du gène G_{TOC}. Ici, les ARNm et les protéines subissent une décroissance exponentielle. Par ailleurs, les photons φ détruisent directement les protéines P_{TOC}.

Les simulations ont été ensuite codées au moyen d'une version « préhistorique » de l'extension IODA pour Netlogo (cf. §3.3.2), d'accès relativement facile à des non-informaticiens.

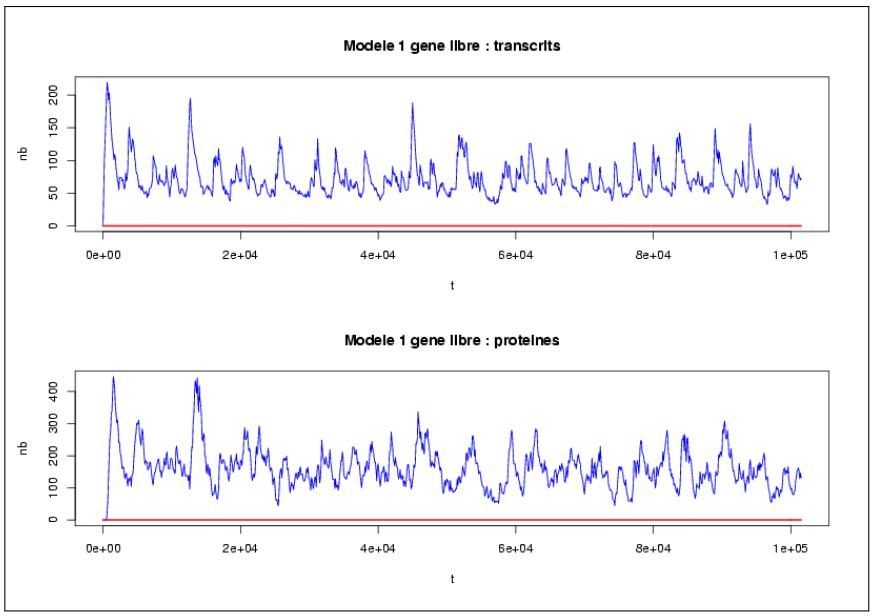


FIGURE 3.4 – Résultats typiques de la simulation à 1 gène du modèle multi-agents, en l'absence de tout signal lumineux : en haut, courbe de l'ARNm ; en bas, celle des protéines. Axe des abscisses : nombre de pas de temps de simulation.

Quelques résultats des simulations

L'étude théorique du modèle à 1 gène avec dégradation enzymatique des protéines [162] prévoit des oscillations pour une plage réduite de valeurs de paramètres. Au contraire, en simulation multi-agents, on obtient très fréquemment des courbes du type de celles de la figure 3.4. Ces oscillations, dont la régularité reste précaire, tiennent à l'aspect spatial du phénomène, absent des modèles équationnels classiques. Par comparaison, les équations à délais [143], qui visent à tenir compte du temps de diffusion des acteurs moléculaires, prévoient des oscillations. Le manque de régularité des pics, en revanche, est lié au caractère stochastique de la fixation d'une protéine sur l'ADN ou de sa libération, et donc à l'initiation ou à l'arrêt de la transcription du gène.

En dépit de cette variabilité, on constate (fig. 3.5) que le système peut être synchronisé par un signal lumineux carré dont la période est la moyenne de l'écart entre les pics de la fig. 3.4. De plus, si l'on passe après la phase d'entraînement jour/nuit en lumière continue comme dans les conditions de laboratoire, les oscillations persistent de façon plus robuste qu'en régime libre.

Quant au modèle à 2 gènes, de nombreux jeux de paramètres conduisent à des oscillations en régime libre, qui peuvent être forcées en alternance jour/nuit, et persistent lors du passage en lumière continue (fig. 3.6). Les déphasages entre les quatre courbes sont proches de ce que l'on peut observer *in vivo*. Ces résultats confortent donc plutôt l'hypothèse d'une boucle de régulation impliquant ces deux gènes.

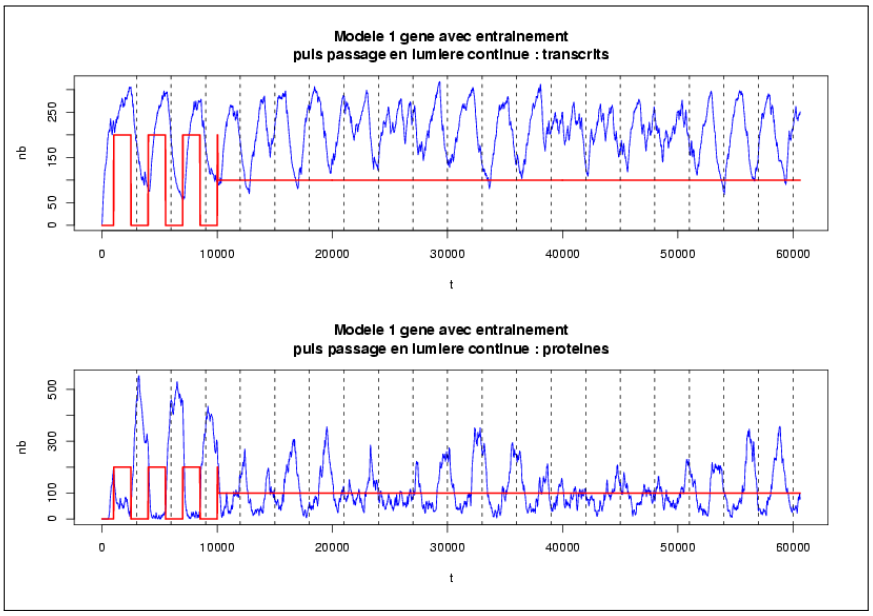


FIGURE 3.5 – Résultats obtenus dans le modèle multi-agents à 1 gène avec l'introduction d'un signal lumineux carré suivi d'un signal lumineux continu. Les pointillés verticaux indiquent la période de forçage.

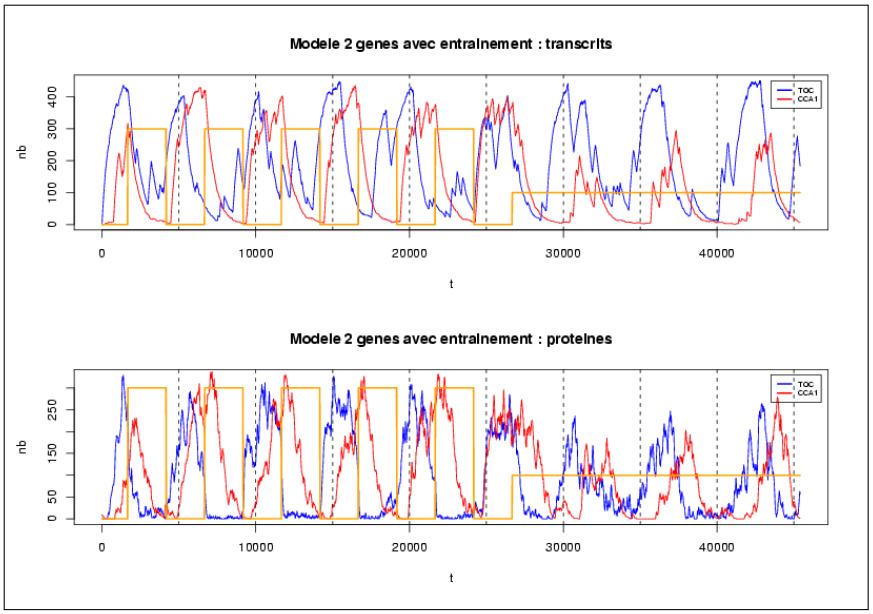


FIGURE 3.6 – Résultats typiques de la simulation à 2 gènes : en haut, courbe des ARNm de TOC et CCA ; en bas, courbe des protéines. La cellule est entraînée d'abord par un signal lumineux carré ($T = 5000$ pas) puis on passe en lumière continue. Ici on a $\delta_{MA} = \delta_{MB} = 1{,}5.10^{-3}$, ce qui équivaut à une demi-vie des ARNm d'un peu plus de 2 h (compatible avec la littérature). À noter : le déphasage progressif en jour continu, le système retournant à sa période libre légèrement supérieure à 24 h.

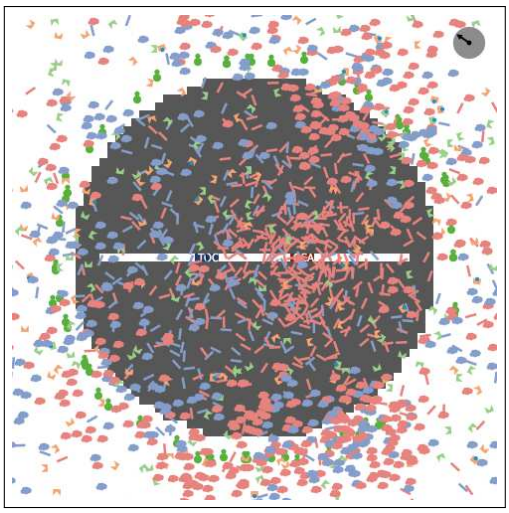


FIGURE 3.7 – Vue du modèle de simulation multi-agents, ici pour un modèle à 2 gènes. **Environnement :** tore plat 2D représentant l'intérieur d'une cellule ; les ribosomes sont à la périphérie du noyau (en grisé). **Agents :** Au centre, un brin d'ADN contenant deux agents gène (TOC en bleu, CCA en rouge) : chacun produit des *ARNm* (bâtonnets) qui diffusent dans la cellule. Lorsque ceux-ci passent à proximité de *ribosomes* (en vert à la périphérie du noyau), ces derniers les traduisent en *protéines* (en bleu). Ces protéines peuvent diffuser se fixer près du gène pour le réprimer, ou disparaître.

3.1.5 Discussion

Dans cet exemple applicatif, la méthode IODA a montré sa capacité à être utilisée par les thématiciens pour l'élaboration d'un modèle conceptuel, et la modification de ce modèle par l'ajout de nouvelles hypothèses quant aux relations entre les acteurs impliqués dans le système. Ainsi par exemple le passage du modèle à 1 gène au modèle à 2 gènes consiste à ajouter des lignes et des colonnes dans la matrice pour les nouvelles familles d'agents, et à placer au bon endroit les interactions envisagées.

Ces modèles peuvent être facilement étendus par l'ajout de nouveaux comportements : par exemple on peut introduire une dégradation enzymatique des protéines dans le modèle à deux gènes, tenir compte de la « marche » des facteurs de transcription le long de l'ADN en ajoutant une interaction de déplacement pour les protéines, etc.

Un des points les plus féconds a été la réflexion sur les hypothèses qui avaient été faites quant à la synchronisation de l'horloge par les signaux lumineux. Nous savions que l'action directe de la lumière sur une des protéines du système était peu vraisemblable, mais il était nécessaire, dans une démarche parcimonieuse et incrémentale, de commencer par là. L'agentification de la lumière au moyen d'agents « photons », qui était en premier lieu une conséquence de l'approche IODA qui veut que toute entité jouant un rôle soit un agent, a permis d'affiner ces hypothèses. On constate en effet que les résultats les plus proches des données expérimentales sont obtenus pour un nombre d'agents « photons » *relativement faible*. Cela suggère une interprétation assez originale de ces agents.

En voici la raison : lorsque la présence de lumière est un « état de l'environnement », les protéines se dégradent à un rythme constant dans le temps durant toute la durée du jour, ce qui donne la « pseudo-réaction » chimique suivante :

$$P \xrightarrow{k_1} \varnothing \qquad \text{d'où la cinétique :} \quad \frac{\mathrm{d}[\mathrm{P}]}{\mathrm{d}t} = -k_1[\mathrm{P}]$$

Au contraire, si l'on considère que les protéines doivent être « touchées » par d'autres entités pour être détruites, l'équation change de forme et devient :

$$P + \varphi \xrightarrow{k_2} \varnothing \qquad \text{d'où une cinétique différente :} \quad \frac{\mathrm{d}[\mathrm{P}]}{\mathrm{d}t} = -k_2[\mathrm{P}][\varphi]$$

On voit clairement que si l'on a un grand nombre d'agents « photons », alors $[\varphi] \quad [\mathrm{P}]$, ce qui redonne la dégradation exponentielle précédente (avec $k_1 = k_2[\varphi]$). En revanche, si le nombre de « photons » est relativement faible, ils ont un *effet limitant* sur la cinétique de dégradation des protéines lorsque ces dernières sont nombreuses, comme dans la dégradation enzymatique de Michaelis et Menten [153]. Aux niveaux d'éclairement utilisés en pratique, il ne peut donc s'agir réellement de photons mais de *photorécepteurs*, c'est-à-dire de molécules présentes en quantité limitée, qui s'activent en présence de lumière et vont ensuite réagir avec les protéines pour les dégrader. Il s'avère que c'était déjà une des hypothèses de travail de l'équipe de Banyuls, non introduite explicitement dans nos modèles : la simulation a donc permis, d'emblée, de conforter la pertinence de nouvelles hypothèses.

3.2 Serious Games : l'humain dans la boucle

Pour être un tant soit peu polyvalent et pouvoir en particulier être utilisé pour la production de jeux vidéo, de simulateurs de conduite, etc., un moteur de simulation doit être en mesure de gérer non seulement des simulations « classiques » n'impliquant que des agents logiciels, mais également des situations dans lesquelles certains agents ont une matérialité extérieure au système : c'est le cas par exemple quand on doit interagir avec des réseaux de capteurs, des robots ou encore pire, des humains.

Ces **simulations participatives** demandent la mise en œuvre de techniques spécifiques pour calibrer le déroulement de la simulation, puisque les agents logiciels doivent impérativement s'adapter au rythme de leurs partenaires physiques. Mais surtout, quand elles impliquent des acteurs humains, elles constituent un banc d'essai redoutable pour tester la *crédibilité* des comportements donnés aux agents logiciels, puisque ces derniers doivent être jugés « vrais » (plus exactement, « vraisemblables ») par les humains qui prennent part à la simulation. Si ce n'est qu'un facteur de succès commercial pour les jeux vidéo, c'est en revanche un point crucial dans les simulateurs de conduite par exemple, où le comportement d'un conducteur ne peut être analysé avec pertinence que si l'environnement où il est immergé est suffisamment réaliste, y compris dans les comportements « déviants » des véhicules simulés [136]. Dans une certaine mesure, on peut donc parler de « test de Turing » [212] *distribué* : les agents artificiels et les agents pilotés par des humains doivent, autant que possible, être indiscernables.

Dans cette partie, nous allons illustrer ce problème à travers le projet Format-Store où l'agent humain est un apprenti vendeur confronté aux demandes de ses clients.

3.2.1 Contexte : le projet Format-Store

☞ `http://www.lifl.fr/SMAC/projects/formatstore/`

Le projet Format-Store a fait l'objet d'un contrat de fin 2009 à fin 2011, financé par le ministère de l'Économie, des Finances et de l'Industrie dans le cadre de l'appel à projet *Serious Game*. Il a été conduit en collaboration avec Idées-3Com, entreprise spécialisée dans la création et la navigation dans des mondes virtuels en 3D, et Enaco, école de commerce, tous deux implantés dans la métropole lilloise. Les images du jeu montrées ci-après sont la propriété de Idées-3Com.

Ce projet a mobilisé à plein temps dans notre équipe un post-doctorant spécialiste des *serious games*, David Panzoli (13 mois), ainsi qu'un ingénieur de Polytech'Lille, Jean-Baptiste Leroy (16 mois), qui ont tous deux travaillé en contact étroit avec les ingénieurs d'Idées-3Com. Il a été en effet nécessaire d'intégrer à la plateforme de visualisation 3D, réalisée en VRML et Javascript, un moteur IODA « allégé », *IODA-light*, qui ne reprend que les fonctionnalités centrales de JEDI pour les adapter à cette architecture logicielle (fig. 3.8).

La mise au point et la validation des comportements des clients simulés se sont appuyées sur les experts en *management* et en *marketing* d'Enaco, ainsi que sur notre propre expérience de simulation dans ce domaine [10]. Je ne mentionnerai ici que les points essentiels du projet, en renvoyant aux articles correspondants pour plus de détails [1, 8, 25, 38]. Format-Store a également fait l'objet d'une démonstration au cours de la conférence PAAMS'2012 [37].

Le but de ce projet était de compléter une formation traditionnelle aux techniques de vente et à la gestion de la relation clientèle par la réalisation d'un *serious game* (ou « jeu sérieux ») immersif, autrement dit une simulation de type jeu vidéo mais dont le caractère ludique doit servir des objectifs pédagogiques balisés. Pour cela, l'apprenant est placé dans des situations, scénarisées ou non, qui le confrontent à des problèmes vus en cours et auxquels il doit apporter des réponses appropriées. Il est impératif que le joueur humain se sente pleinement impliqué dans la simulation, sans quoi il peut relâcher son effort et ne pas chercher à satisfaire toutes les demandes ; il faut également que ce qu'il voit lui semble familier et réaliste, sous peine de ne pas identifier correctement les situations à problème ; enfin, le jeu doit présenter des difficultés modulées en fonction des performances du joueur afin de le maintenir dans une attitude active sans le submerger.

Le cadre général retenu dans Format-Store est celui d'un apprenant placé à travers son avatar (fig. 3.9) dans la situation d'un vendeur d'un magasin de détail de superficie moyenne (de type épicerie ou supérette). Pour réaliser cette simulation, il a fallu définir non seulement les comportements que devaient manifester les clients simulés et les autres entités présentes dans le magasin (articles, caisses, panneaux de signalisation, éléments de structure comme les rayonnages, etc.), mais également les comportements « corrects » attendus de la part du joueur afin de pouvoir l'évaluer. Outre les tâches quotidiennes (mise en rayon, vérification des dates de péremption,

nettoyage, élimination des emballages, commande d'articles en rupture, etc.), le vendeur est confronté à des situations-problèmes, qui se présentent en pratique sous la forme de dialogues interactifs avec un client. La mise en contexte des apprentissages classiques est donc enrichie par la nécessité, dans le jeu, de prioriser ces diverses tâches en trouvant des compromis entre la qualité des réponses apportées ou des actions engagées et la gestion du temps disponible.

3.2.2 *Serious Games* et SMA

L'immersion du joueur dans un *serious game* résulte habituellement de la confluence entre le réalisme visuel du jeu et la scénarisation des interactions avec les personnages non-joueurs (PNJ). L'accent peut être mis tantôt sur le premier [63, 169], auquel cas les PNJ ont des capacités comportementales extrêmement réduites (comme par exemple transmettre des informations), tantôt sur le second en réduisant drastiquement la liberté du joueur à travers des scénarios scriptés assez rigides.

La simulation multi-agents permet donc *a priori* de restaurer un équilibre entre ces deux pôles, en associant au réalisme visuel d'agents représentés dans un monde 3D une richesse comportementale qui vient de leur autonomie et de leur capacité d'adaptation. En outre, comme nous allons le voir, l'approche IODA permet d'introduire facilement une scénarisation des comportements, et d'enrichir le jeu *ad libitum* en fonction des besoins.

Toutefois, la difficulté de la réalisation d'un *serious game* immersif au moyen d'un SMA réside dans la poursuite simultanée de plusieurs objectifs (fig. 3.10) : l'immersion dans un monde simulé réaliste, le maintien de l'attention et de la motivation par les aspects ludiques et la transmission de contenus pédagogiques — et donc dans la résolution conjointe des problématiques spécifiques de chacun de ces axes. En tant que jeu dans lequel l'apprenant doit être immergé, le *serious game* immersif introduit les contraintes suivantes :

1. La simulation doit être *participative*, autrement dit prendre en compte les différences de modalité et de temporalité entre l'humain et les PNJ. Pour autant, puisqu'il s'agit aussi d'une simulation, le joueur humain doit pouvoir être remplacé par un agent autonome : c'est d'ailleurs une façon de valider les comportements des PNJ.
2. Le comportement des agents doit être perçu par le joueur humain (ou par d'autres observateurs) comme *réaliste et cohérent* pour favoriser l'immersion dans le jeu.
3. Le comportement des agents doit *s'adapter* non seulement à un environnement dynamique mais aussi aux actions du joueur, relativement imprévisibles.

De plus, les contenus pédagogiques doivent être intégrés à la simulation à travers les comportements des agents :

4. Les comportements des agents doivent être définis de façon *modulaire* pour gérer des capacités de complexités différentes : se déplacer, interagir, raisonner, communiquer, etc.
5. Pour faciliter les échanges avec les experts, la formulation des comportements doit être *explicite et intelligible*.

6. La conception de la simulation étant essentiellement incrémentale et sa validation empirique (par l'observation des comportements qui se déroulent et leur modification), le modèle doit être aisément *révisable*.

Enfin, le seul fait de réaliser un jeu à contenu pédagogique entraîne aussi des contraintes :

7. L'apprenant est *évalué* durant sa session de jeu, selon des modalités qui peuvent être extrêmement diversifiées et doivent pouvoir être combinées.
8. La transmission des contenus pédagogiques doit être *scénarisée*, mise en contexte dans le jeu pour que l'apprenant se sente impliqué : cela suppose de pouvoir *susciter des situations* d'une façon relativement contrôlée en dépit de l'autonomie laissée aux agents.

Comme nous allons le voir, l'approche orientée interactions permet de répondre à ces diverses contraintes.

3.2.3 Mise en œuvre par approche orientée interactions

Comportement des clients virtuels

La spécification des comportements des clients simulés a été menée en collaboration avec les experts d'Enaco. En règle générale, chaque client entre dans le magasin avec une liste de courses qu'il s'efforce de satisfaire : pour cela il se déplace de façon autonome à la recherche de ses produits et les place dans son panier. Lorsqu'il a terminé ses achats, il se dirige vers les caisses, puis sort du magasin. On peut en outre lui affecter une situation-problème qui va lui demander de trouver le vendeur et d'engager un dialogue : dans cette situation (fig. 3.11), le joueur doit sélectionner les bonnes réponses aux questions du client, après quoi le client poursuit son activité. Toutefois, sa patience est limitée, de sorte que s'il ne trouve pas ses articles, ou s'il n'est pas satisfait des réponses du vendeur ou de l'état du magasin, il abandonne ses achats et sort.

Construction du modèle

Outre les clients simulés, le magasin est également peuplé de nombreux autres agents, en vertu de la méthode IODA selon laquelle *toute entité* doit être représentée par un agent. Or, parmi les situations-problèmes non dialogiques, on trouve des perturbations introduites suite aux actions du vendeur (encombrement d'une allée après déballage d'articles remis en rayon) ou à d'autres événements (rupture d'approvisionnement d'un article, péremption, panne d'un éclairage, allée souillée par la chute d'un article, etc.).

Pour les experts, la participation à la modélisation a donc consisté essentiellement à recenser toutes les entités impliquées dans le magasin et à définir leurs interactions, ce qui n'a pas posé de difficulté. La formulation du comportement des agents sous forme d'interactions, c'est-à-dire de règles, est très naturelle en marketing et son placement dans la matrice intuitif (tab. 3.3).

Les groupes d'agents ainsi identifiés sont donc les suivants :

— le Vendeur ;

— les Clients ;
— les Articles ;
— les panneaux (Affichage) utilisés pour orienter les clients dans le magasin ;
— les Portes (qui créent ou éliminent des clients) ;
— les Caisses ;
— les files d'attente (FileAttente) engendrées par ces caisses ;
— les emballages utilisés chaque fois qu'un article est remis en rayon (Carton) ;
— les Taches qui résultent de certaines interactions et doivent être nettoyées.

Bon nombre de ces agents, qui dans des approches de conception multi-agents classiques seraient implémentés au moyen d'autres types d'entités (par exemple des *objets*, des *ressources*, des *artefacts*, etc.) ont en fait un rôle actif à jouer. Par exemple, plutôt que de considérer que les clients cherchent des informations dans leur environnement pour s'orienter, ce sont les affichages qui *informent* les clients (avec des conditions liées entre autres à l'orientation des agents).

L'intégralité du comportement de chaque agent est ainsi représenté de manière extrêmement *intelligible* (point 5 de la fig. 3.10). En outre, on constate que beaucoup d'interactions sont suffisamment génériques pour être utilisées par plusieurs familles d'agents. Cela tient bien entendu au *polymorphisme* des primitives à partir desquelles elles sont écrites, et qui sont implémentées de façon différenciée dans les diverses familles d'agents.

La matrice d'interaction a été élaborée de façon incrémentale et constitue un modèle très simplement *révisable* (point 6) par ajout, suppression, déplacement, changement de paramètres (priorité ou distance) des interactions, chaque modification pouvant être testée immédiatement sans réécriture de code.

Qu'en est-il des autres points soulevés par la réalisation d'un *serious game* immersif au moyen d'un SMA ? La *modularité des comportements* (point 4) découle directement de leur externalisation sous la forme d'interactions définies de façon indépendante, et qui peuvent avoir une complexité arbitraire.

La *scénarisation* (point 8), autrement dit l'intégration des contenus pédagogiques au sein du jeu, que ce soit dans le comportement des PNJ ou dans des événements, est également facilitée par l'approche orientée interactions : soit par des dialogues qui constituent une interaction parmi d'autres, soit par la succession d'interactions qui se déclenchent en cascade du simple fait que leurs conditions ou déclencheurs deviennent valides. Par exemple, si un client tente d'effectuer Prendre sur un article et le lâche (ce qui se produit avec une probabilité dépendant du niveau de jeu), l'article peut effectuer GénérerTache, et le nouvel agent Tache pourra soit subir Nettoyer de la part du Vendeur, soit effectuer Contrarier sur les autres Clients.

Réalisme des comportements

Comme l'a signalé Kubera [134], l'approche IODA permet, sans chercher à en donner une implémentation *stricto sensu*, de se rapprocher du concept d'affordances. D'après Gibson [119], les fonctions possibles d'un objet dans son environnement (donc ses capacités d'interaction) peuvent être interprétées comme étant suggérées par l'objet lui-même, selon sa forme, sa position, etc. Cette théorie est fréquemment utilisée dans les jeux vidéo et *a fortiori* dans la conception d'interfaces ergonomiques.

Comme *toutes les entités sont des agents*, il est tout à fait naturel de doter des « objets » de la capacité à effectuer des interactions et non seulement à en subir.

Par ailleurs, pour ce qui est des agents « humanoïdes », la priorité, les déclencheurs et conditions des interactions qu'ils peuvent effectuer ont été définis de telle sorte que, sans se succéder de façon déterministe, ces interactions ont lieu selon un ordre logique. Ainsi les clients simulés semblent suivre un plan et sont bien perçus comme dotés de *comportements réalistes* (point 2), impression renforcée par le fait chaque client simulé est doté de sa propre liste de courses, d'une connaissance plus ou moins précise de l'organisation spatiale du magasin, et de capacités de mémorisation et d'apprentissage qui lui permettent de s'orienter selon un parcours vraisemblable.

Ce même mécanisme garantit également leur adaptativité (point 3) et leur diversité. En effet, la capacité des clients à trouver les articles de leur liste de courses ne dépend pas de leur connaissance préalable du magasin : les produits peuvent donc être déplacés, retirés, les allées obstruées, sans affecter leur capacité à faire leurs achats.

Les clients, qui sont différenciés par leur apparence visuelle, le sont aussi par les profils comportementaux qui leur sont donnés, notamment par l'affectation de listes de courses différentes, de niveaux variables de patience, outre bien sûr les situations-problèmes dont ils peuvent être les vecteurs. Ces capacités d'adaptation des PNJ et la diversité de leur comportement sont essentielles à la *crédibilité* de la simulation, donc à l'immersion du joueur humain.

L'humain dans la boucle

Le *serious game* étant également une *simulation participative* (point 1), les agents logiciels partagent leur environnement avec l'apprenant, qui doit donc être intégré de façon non bloquante. Dans Format-Store, comme le montre la matrice d'interaction (tab. 3.3), le Vendeur est bien représenté par un agent : celui-ci est couplé à l'avatar de l'apprenant dans le jeu. Lorsque le joueur commande son avatar, ses ordres sont envoyés à l'agent par un système de boîte à lettres qui se substitue au mécanisme ordinaire de sélection d'interaction, de sorte que l'inaction du joueur laisse simplement la simulation suivre son cours normalement. Du reste, une des étapes de la validation a consisté à laisser tourner la simulation sans vendeur.

Il en découle également que le joueur humain, par ce procédé, peut si nécessaire contrôler n'importe quel autre agent de la simulation (par exemple on peut envisager, même si cela ne correspondait pas aux objectifs de Format-Store, qu'un enseignant prenne le contrôle d'un Client).

En tant qu'outil pédagogique, la simulation doit permettre une *évaluation flexible* (point 7) de la performance de l'apprenant. Cette évaluation repose en l'occurrence sur deux critères : le premier est calculé à l'issue de chaque dialogue avec un client (en fonction de la pertinence des réponses fournies), et la diversité des comportements et des événements garantit un large éventail de situations d'évaluation ; le deuxième intègre le niveau de satisfaction de chaque client qui sort du magasin, ce qui dépend de sa capacité à avoir trouver ses articles et des interactions qui peuvent avoir affecté sa patience (saleté du magasin, allées encombrées, etc.). Les paramètres qui gèrent les flux de clients entrants et la probabilité des événements sont adaptés en fonction du score du joueur [1].

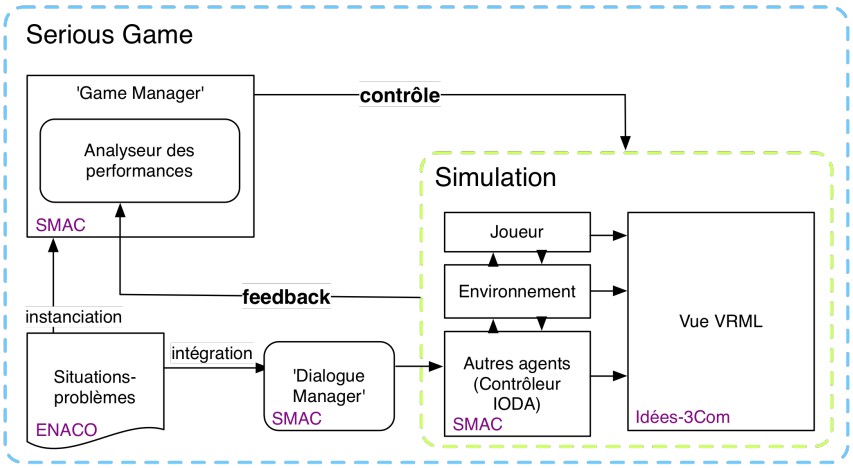


FIGURE 3.8 – L'architecture logicielle de FORMAT-STORE. Pour la partie simulation, l'animation 3D a été assurée par Idées-3Com ; les intervenants de l'équipe SMAC ont géré toute la partie comportementale, ainsi que l'intégration des dialogues correspondant aux situations-problèmes identifiées par les experts, et le contrôle adaptatif de la difficulté du jeu (source : David Panzoli — rapport interne).

FIGURE 3.9 – Dans FORMAT-STORE, le joueur est placé dans un magasin *via* un avatar contrôlé au clavier ; il peut aussi interagir avec les clients et les articles en utilisant la souris.

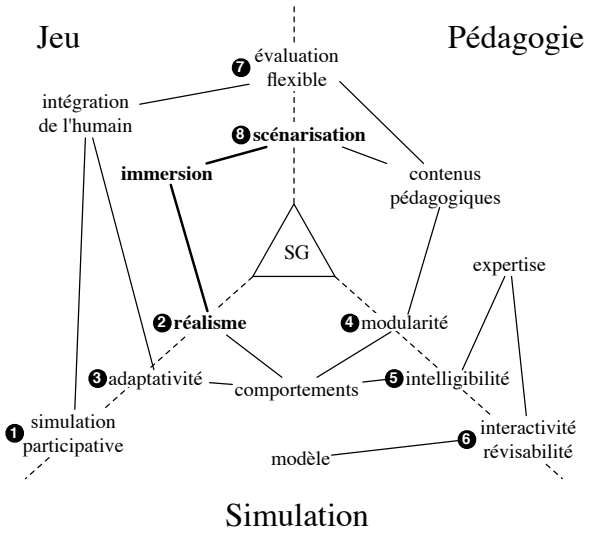


FIGURE 3.10 – Interdépendance des problèmes soulevés par un *serious game* immersif. Les trois composantes d'un tel jeu (pédagogie, jeu, réalisme) engendrent des problématiques qui leur sont propres, mais dont la résolution est compliquée par la nécessité de maintenir une équilibre entre les trois objectifs (source : [25]).

FIGURE 3.11 – Les dialogues de situations-problèmes donnent lieu à des interactions conversationnelles, par le biais d'une interface spécifique.

Cibles / Sources	∅	Vendeur	Client	Porte	Affichage	Caisse	FileAttente	Article	Tache	Carton
Vendeur			Dialoguer (1 ; 0)					Ôter (5 ; 0) Réapprovisionner (5 ; 0) Ranger (5 ; 0)	Nettoyer (3 ; 0)	Ranger (3 ; 0)
Client	Errer (0) AllerVers (1)	Repérer (10 ; 9) Patienter (2 ; 3)		Sortir (1 ; 8)		Payer (5 ; 6)	Entrer (2 ; 4) SePlacer (1 ; 5) Sortir (1 ; 7)	Prendre (5 ; 2)		
Porte	GénérerClient (1)		Informer (10 ; 0)							
Affichage			Informer (10 ; 0)							
Caisse			Informer (10 ; 0) Encaisser (2 ; 1)							
FileAttente										
Article	Périmer (3) GénérerTache (4) GénérerCarton (4)		Informer (10 ; 0) Contrarier (1 ; 1) AvertirRupture (1 ; 2)							
Tache			Contrarier (1 ; 0)							
Carton			Contrarier (1 ; 0)							

Table 3.3 – Matrice d'interaction utilisée dans Format-Store. On voit que des agents habituellement considérés comme objets, ressources ou artefacts jouent en fait un rôle actif, p. ex. Tache ou Article (cf. [25]).

3.2.4 Bilan

Tout au long du déroulement de ce projet, l'approche IODA a montré son intérêt tant en ce qui concerne la méthodologie de conception de modèles de simulation, construits de façon incrémentale sous une forme explicite qui peut faire l'objet de discussions avec les experts et de remaniements peu coûteux, que pour ce qui est de la réalisation d'une simulation capable d'intégrer des joueurs humains et d'interagir avec une architecture logicielle tierce en charge du monde virtuel.

Les *serious games* immersifs induisent des problématiques couplées de pédagogie, de jeu et de simulation, qui sont résolues assez simplement du seul fait de la séparation déclaratif/procédural opérée par IODA. Par ailleurs le choix de faire de toute entité un agent s'avère une fois de plus très naturel et facilite l'explicitation des connaissances des experts, en se rapprochant de théories psychologiques comme celle des affordances.

3.3 Diffusion de l'approche orientée interactions

Outre des applications concrètes répondant à des problématiques spécifiques à tel ou tel domaine, nous nous sommes efforcés de diffuser autant que possible notre démarche à la fois dans la communauté multi-agents et chez des thématiciens qui utilisent des modèles centrés individus, et d'en montrer les avantages.

Je présenterai ici deux actions notables dans ce sens : premièrement, le projet Galaxian, qui a débouché sur la réalisation d'une démonstration capable de tourner pendant des heures dans une salle de réalité virtuelle, tout en permettant d'expliquer le fonctionnement de simulations orientées interactions ; deuxièmement, l'extension IODA pour la plateforme NetLogo, qui permet de toucher une large communauté de concepteurs de modèles à base d'agents et d'enseigner aisément nos concepts soit à des thématiciens, soit à des étudiants en informatique.

3.3.1 Le projet Galaxian

☞ `http://www.lifl.fr/SMAC/projects/galaxian`

Contexte

Le projet Galaxian [36] a été réalisé en 2011–2012 au sein de l'équipe SMAC, en réponse à un appel à démonstration du LIFL, dans le cadre de la plateforme PIRVI [1] (Plateforme Interactions-Réalité Virtuelle-Images) de l'IRCICA à Lille. Il a pour objectif de montrer la mise en œuvre de la méthode de simulation IODA dans un cadre applicatif ludique : en l'occurrence, une bataille 3D entre divers types de vaisseaux spatiaux.

Le simulateur de démonstration de bataille a été implémenté en C# au sein du moteur de jeu professionnel Unity3D [2] par Marc-Antoine Dupré (ingénieur contractuel

1. `http://www.lifl.fr/pirvi/`
2. `http://unity3d.com/`

INRIA) et David Panzoli (post-doctorant), à partir d'une implémentation d'un moteur de simulation IODA que j'avais réalisé pour Unity3D. Il est capable d'animer une bataille spatiale en continu, sans répétitions, durant des dizaines d'heures, et tourne sur un mur-écran de 6m de large avec un son spatialisé. Par ailleurs il encapsule son propre matériel pédagogique sous forme d'une part de diapositives et d'animations, d'autre part d'informations de contrôle (« *gizmos* ») qui peuvent être ajoutées à la vue de la bataille : cela permet une présentation impromptue du fonctionnement du moteur de simulation lors des démonstrations. Enfin, grâce à la portabilité de Unity3D, la démonstration est également disponible en ligne à partir de la page web de l'équipe.

Présentation

La bataille oppose deux équipes, l'une blanche, l'autre noire, chacune composée de plusieurs familles d'agents, dont certaines leur sont spécifiques :

— les **chasseurs** (Fighter) blancs sont les unités combattantes de l'équipe blanche et affrontent les chasseurs noirs, et vice-versa — ces chasseurs ont un très large halo de perception ;

— les **croiseurs** (Cruser) blancs (resp. noirs) envoient des fournées de chasseurs blancs (resp. noirs) à l'assaut de l'équipe adverse, lorsque le nombre de chasseurs est tombé en-dessous d'un certain seuil — dans la démonstration ces bâtiments sont indestructibles afin de garantir que de nouveaux chasseurs peuvent être relancés en permanence ;

— les **frégates** (Frigate), propres à l'équipe noire, sont des vaisseaux indestructibles, de taille intermédiaire, armés de tourelles puissantes destinées à repousser les chasseurs ennemis — elles ne peuvent pas être attaquées par des chasseurs isolés ;

— les **escadrilles** (ou **escouades**, Squad) sont propres à l'équipe blanche : elles sont formées graduellement par des chasseurs qui décident de se regrouper, puis attaquent les frégates, avant de se disperser soit parce qu'elles ont terminé leur raid, soit parce que trop de chasseurs ont été abattus — en pratique dans la simulation, les escadrilles existent bel et bien en tant qu'agents mais ne sont pas représentées par un objet graphique du moteur de jeu ; on peut les visualiser en « vue augmentée » sous la forme d'une sphère plus grande que celle des autres agents.

Les comportements de ces agents ont été définis *via* la matrice d'interaction ci-dessous (tab. 3.4) :

Afin de suivre les détails des comportements, il est possible de visualiser certaines informations, telles que les relations de perception entre agents, ou les relations source-cible qui donnent lieu a une interaction (fig. 3.12).

Bilan

Outre son fort impact [3] en tant que démonstration, le projet Galaxian a été l'occasion de montrer plusieurs qualités de l'approche IODA. D'une part, les algorithmes que nous avons développés pour la simulation orientée interactions sont capables

3. Galaxian a obtenu le prix IBM de la meilleure démonstration à la conférence PAAMS'2013.

Sources \ Cibles	∅	BFighter	WFighter	BFrigate	WSquad
BFighter	(ChercherCible ; 0) (Tirer ; 8) (Exploser ; 9)		(Affronter ; 2 ; 1000) (Intercepter ; 3 ; 1000) (Engager ; 4 ; 1000) (Fuir ; 5 ; 1000)		
WFighter	(ChercherCible ; 0) (Tirer ; 8) (Exploser ; 9)	(Affronter ; 2 ; 1000) (Intercepter ; 3 ; 1000) (Engager ; 4 ; 1000) (Fuir ; 5 ; 1000)	(CréerEscouade ; 2 ; 30)		(Rejoindre ; 6 ; 100) (Suivre ; 7 ; 1000)
BFrigate					(Riposter ; 0 ; 200)
BCruser	(LancerChasseur ; 0)				
WCruser	(LancerChasseur ; 0)				
WSquad	(Disparaître ; 0) (Tirer ; 3)			(Engager ; 1 ; 5000)	(Fusionner ; 5 ; 50)

TABLE 3.4 – Matrice d'interaction utilisée pour la définition des comportements des agents dans GALAXIAN.

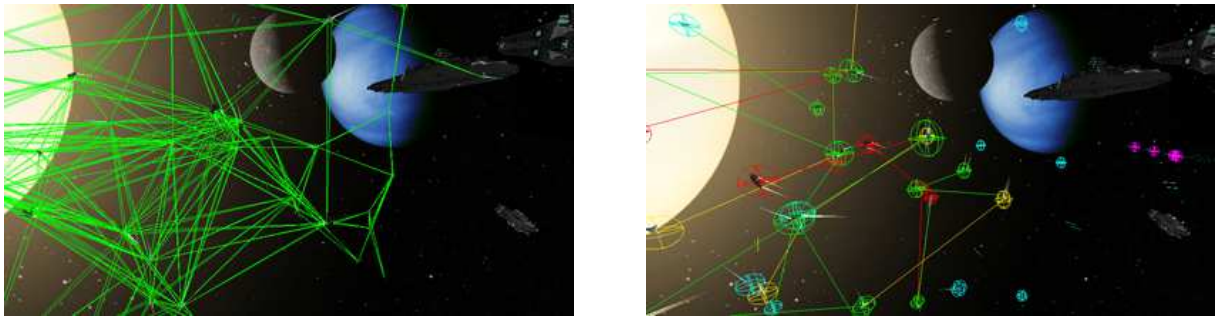


FIGURE 3.12 – Deux vues augmentées de la simulation GALAXIAN : à gauche, les traits représentent les relations de perception entre agents ; à droite, la couleur des sphères autour des agents code l'interaction qu'ils effectuent et le trait les relie à leur cible.

de s'insérer dans un moteur de jeu professionnel sans perte de performance, et permettent une dissociation entre la gestion des comportements de haut niveau (choix des interactions par les agents) par le moteur IODA, à pas de temps fixes, et la gestion des événements graphiques (collisions, mouvements) par le moteur de jeu qui tourne avec un *framerate* variable.

D'autre part, l'approche orientée interactions a été utilisée pour définir des comportements d'agrégation d'agents en groupes eux-mêmes dotés de comportements propres.

Enfin, en tant que méthode de conception de simulations, la méthode IODA s'est particulièrement bien prêtée à une conception incrémentale des comportements, en commençant par exemples par le réglage des interactions de navigation et de poursuite (ChercherCible, Affronter, Engager, Fuir) pour s'assurer d'une couverture homogène de la zone de bataille et du placement de la caméra, puis l'ajout d'interactions de combat (Tirer, Exploser, LancerChasseur, etc.) calibrées pour maintenir une population équilibrée de chasseurs de chaque camp ; enfin, les comportements de groupe (CréerEscouade, Rejoindre, Suivre, Disparaître) ont été introduits après validation (empirique et visuelle) des comportements précédents. La simulation peut ainsi être complexifiée et enrichie étape par étape de façon très modulaire.

3.3.2 IODA pour NetLogo

> *The only way to get rid of a temptation is to yield to it.*
>
> WILDE [215]

☞ `http://www.lifl.fr/SMAC/projects/ioda/ioda_for_netlogo/`

Contexte

La plateforme de simulation multi-agents NetLogo, développée par WILENSKY [217] et son équipe au *Center for Connected Learning and Computer-Based Modeling* (CCL) de la Northwestern University, est (avec Repast [166]) l'une des plus utilisées, tout particulièrement en-dehors de la communauté des informaticiens. Son utilisation est en effet particulièrement aisée, car elle fournit un environnement intégré comprenant des outils de construction d'interface graphique, d'analyse statistique, de planification d'expériences, d'exportation sous forme d'*applet* etc., le tout programmable dans un langage d'un assez haut niveau d'abstraction[4] avec une syntaxe simple fonction-arguments (en fait dérivée de LISP — sans les parenthèses !). Il est possible de définir

4. Parmi les primitives de NetLogo on trouve par exemple des fonctions permettant de calculer un gradient dans l'environnement ou demandant à tous les agents d'effectuer une procédure de façon concurrente.

des familles d'agents (*breeds*) en plus de celles par défaut : les « tortues » (*turtles*), les *patches* (discrétisation de l'environnement en carrés unitaires) et les liens entre agents (*links*). En outre, elle est distribuée avec une documentation détaillée, des tutoriels et plus d'une centaine d'exemples de modèles issus de nombreuses disciplines (biologie, physique, chimie, sciences sociales, économie, etc.). Enfin, elle garantit une reproductibilité totale des résultats de simulation à graine aléatoire fixée, ce qui est évidemment un critère de scientificité crucial.

D'un point de vue épistémologique, NetLogo s'inscrit dans le prolongement de la démarche constructionniste de Papert [170], inventeur du langage LOGO, et de son disciple Resnick [179] qui est l'auteur de StarLogo, version parallèle (multi-tortues) du LOGO et précurseur de NetLogo. L'objectif n'est pas de fournir une plateforme dotée d'une puissance de calcul redoutable, mais bel et bien de permettre à des thématiciens d'exprimer leur modèle de façon simple et d'en tester des prototypes pour avoir un retour rapide sur leurs hypothèses.

Il n'en reste pas moins, comme nous l'avons déjà signalé [13, 12, 3], que la programmation correcte (i.e. sans erreur susceptible d'introduire des biais dans les résultats) d'un moteur de simulation est loin d'être aisée pour un thématicien mal formé à l'algorithmique, voire pour certains informaticiens.

La tentation à laquelle nous avons succombé était donc grande, de profiter de l'audience de NetLogo pour lui « greffer » un moteur IODA, en vue d'une part, de simplifier et d'uniformiser la conception de modèles en séparant le déclaratif du procédural, et d'autre part, de réduire le code à écrire à des procédures ou fonctions de petite taille (les primitives de perception et d'action des agents).

J'ai mis au point les premières versions de l'extension *IODA pour NetLogo* en 2007 (notamment pour la simulation de l'horloge d'*Ostreococcus tauri* (§3.1.1), puis l'architecture a été entièrement revue courant 2011 pour en améliorer les performances et profiter des améliorations offertes par les nouvelles versions de NetLogo (4.1.3 puis 5) ; je présenterai donc ici la version 2 de cette extension qui est celle distribuée depuis début 2012 (v. 2.2 depuis octobre 2013).

Architecture

La plateforme NetLogo s'appuie sur la machine virtuelle Java, de sorte que le langage utilisé pour la programmation des modèles dans la plateforme est d'abord *byte-compilé*. L'API de la plateforme est ouverte et (légèrement) documentée. Pour étendre les commandes et structures de données disponibles, on peut procéder de deux façons : soit écrire un fichier séparé dans le langage NetLogo et l'inclure dans les modèles qui en ont besoin, soit écrire ses propres classes Java et les empaqueter au sein d'une archive JAR.

Après les premières versions exclusivement en NetLogo, faciles à maintenir mais peu performantes, j'ai décidé de suivre une voie hybride : les structures de données (définition des interactions, matrices d'interaction et de mise à jour) et leurs algorithmes propres (notamment lecture et analyse des fichiers de configuration) ont été écrits directement en Java, tandis que le moteur IODA proprement dit, susceptible d'être modifié pour des besoins différents, a été écrit comme fichier d'inclusion en NetLogo, qui définit par ailleurs diverses politiques concernant l'ordonnancement des agents ou la sélection des cibles des interactions.

Cela permet d'écrire le code d'un modèle de simulation en IODA-NetLogo à partir d'un gabarit (fig. 3.13), à savoir :

— les clauses d'inclusion de code et de chargement de l'extension ;
— la procédure d'initialisation `setup` qui se termine par la lecture des fichiers de configuration définissant les interactions et les matrices ;
— la procédure `go` exécutant chaque pas de temps de la simulation, qui délègue l'exécution à une procédure de l'extension (`ioda:go`).

Après quoi, il reste à écrire le code de chacune des primitives utilisées par chaque famille d'agents : il s'agit de fonctions (*reporters*) ou de procédures NetLogo portant le nom de la primitive préfixé par le nom de la famille d'agents.

model.nlogo

```
__includes ["IODA_2_2.nls"]
extensions [ioda]

to setup
  clear-all
  ioda:load-interactions "interactions.txt"
  ioda:load-matrices "matrix.txt" " \t()"
  ioda:setup
  reset-ticks
end


to go
  ioda:go
  tick
end
```

FIGURE 3.13 – Le gabarit utilisé pour l'écriture des modèles de simulation en IODA-NetLogo.

Selon la méthodologie descendante de IODA, mieux vaut commencer par décrire les interactions réalisables, ainsi que la matrice d'interaction et la matrice de mise à jour. Pour cela, deux fichiers texte de configuration doivent être définis :

— l'un pour décrire les interactions : nom, déclencheur, condition et actions (fig. 3.14) ;
— l'autre pour définir les matrices d'interaction et de mise à jour, au format CSV (fig. 3.15).

Dans l'exemple donné ici (fig. 3.15), la première ligne hors commentaires du fichier (`Ants Take (30) Food (1) BEST:quality`) s'interprète de la façon suivante : la famille d'agents Ants peut effectuer l'interaction Take sur des agents de la famille Food, avec une priorité de 30, à une distance maximale de 1, et en utilisant une politique de sélection de cible qui maximise la primitive `quality` de la cible. Comme l'interaction Take utilise les primitives `can-take-load?` et `take-target` pour la source et `die` pour la cible, il faut donc déclarer dans le code NetLogo les deux familles `ants` et `food`, et écrire :

— une procédure `ants::filter-neighbors` pour définir le halo de perception des foumis (agents actifs) ;

interactions.txt

```
interaction Follow
  trigger       foraging?
  actions       follow-target  target:decrease-strength
end

interaction Take
  condition     can-take-load?
  actions       take-target  target:die
end

interaction MoveRandomly
  actions       wiggle
end

interaction ReturnToNest
  trigger       carrying-food?
  actions       drop-pheromone  move-to-home
end

interaction DropFood
  condition     carrying-food?
  actions       drop-load  turn-back
end

interaction Die
  trigger       too-weak?
  actions       die
end

interaction Evaporate
  actions decrease-strength
end

interaction UpdateView
  actions update-label
end

interaction Steal
  trigger       foraging?
  condition     target:carrying-food?
  actions       steal-load
end
```

FIGURE 3.14 – Fichier texte de définition des interactions. L'exemple est tiré de la réécriture d'un modèle classique de NetLogo, la recherche de nourriture par des fourmis avec constitution de pistes [216]. Chaque déclencheur ou condition est constitué d'une conjonction de primitives booléennes, et la partie actions d'une séquence de primitives.

matrix.txt

```
; FILE FORMAT:
; Source        Interaction (priority) [UPDATE]
; or
; Source        Interaction (priority) Target (distance) [<TARGET-SELECTION-POLICY>]

; interactions performed by ants
Ants    Take          (30)       Food  (1)       BEST:quality
Ants    DropFood      (30)       Nests (1)
Ants    Follow        (20)       Food  (5)       BEST:quality
Ants    Follow        (10)       Pheromones (5)  BEST:strength
Ants    ReturnToNest (10)
Ants    MoveRandomly (0)

; interactions performed by pheromones
Pheromones      Die           (20)
Pheromones      Take          (10)       Pheromones (0.8)
Pheromones      MoveRandomly (0)
Pheromones      Evaporate     (10)       UPDATE

; interactions performed by nests
Nests           UpdateView     (0)       UPDATE
```

FIGURE 3.15 – Fichier CSV définissant les matrices d'interaction et de mise à jour. Même exemple que précédemment.

- une fonction (*reporter*) `ants::can-take-load?` qui teste si la fourmi peut prendre une charge ;
- une procédure `ants::take-target` pour charger la fourmi avec sa cible ;
- une procédure `food::die` pour éliminer le morceau de nourriture prélevé ;
- une fonction `food:quality` qui donne une évaluation de la qualité de la nourriture.

Le code correspondant est donné ci-après (fig. 3.16). Le principal avantage est le découpage d'un code habituellement « lourd » (avec des branchements explicites pour gérer les diverses situations) en des comportements explicites sous forme texte et de petits morceaux de code indépendants. On procède ainsi pour chaque nouvelle ligne ajoutée dans la matrice d'interaction ou dans la matrice de mise à jour. Par souci de commodité nous avons défini dans l'extension une procédure qui donne le patron des primitives nécessaires qui ne sont pas encore écrites.

Si l'on compare le modèle « *Ants* » originel de NetLogo à notre simulation, outre cette simplification du code les phéromones et la nourriture ont été agentifiées et peuvent ainsi être dotées de caractéristiques propres susceptibles d'enrichir le modèle, comme par exemple la qualité de la nourriture par exemple, ou encore un déplacement anisotrope des phéromones sous l'effet du vent, etc. De plus, dans l'exemple complet fourni en tant que tutoriel, on peut tester le modèle avec des matrices d'interaction différentes, autorisant des variantes comportementales comme la « capture » de nourriture sur le modèle de robots « dockers » [97].

Nous avons également implémenté plusieurs politiques de sélection de cible d'usage courant, comme par exemple `ALL` (appliquer l'interaction à toutes les cibles

ajouts dans `model.nlogo`

```
; définition des familles d'agents
breed [ants ant]
breed [food]

; définition de leurs attributs
ants-own [carrying? motivation quality]
food-own [ quality]

; définition de la perception des agents
to ants::filter-neighbors
  ioda:filter-neighbors-in-radius 5
end

; définition des primitives nécessaires
; pour la 1e ligne de la matrice d'interaction
to-report ants::can-take-load?
  report not carrying?
end

to ants::take-target
  set motivation 1
  set carrying? true
  set color red
  set quality [quality] of ioda:my-target
end

to food::die
  ioda:die
end

to-report food::quality
  report quality
end
```

FIGURE 3.16 – Les ajouts dans le code du modèle NetLogo correspondant 1° à la définition des familles d'agents, de leurs propriétés et de leur perception, et 2° à l'implémentation des primitives nécessaires pour la première ligne du fichier `matrix.txt`.

qui vérifient les conditions), `BEST:prop` (choisir la cible qui maximise la primitive `prop`), `NUMBER:n` (choisir obligatoirement n cibles), etc.

Utilisation et distribution

Cette extension est accompagnée d'un dictionnaire des fonctions et procédures disponibles, et surtout d'un tutoriel détaillé avec une douzaine d'exemples de complexité croissante pour permettre une prise en main graduelle des concepts de l'approche orientée interaction et de sa mise en œuvre dans NetLogo.

Depuis début 2012, ce paquet est diffusé en téléchargement libre sous licence GPL 3. Il est destiné en première intention aux thématiciens familiers de NetLogo qui souhaiteraient rendre les comportements de leurs modèles plus lisibles (en pratique, moins dépendants du code) et plus facilement révisables. Une fois les interactions écrites, il suffit de commenter ou de démasquer une ligne de la matrice pour changer complètement le modèle ; de plus des fichiers de configuration différents peuvent être écrits et chargés selon l'expérience à effectuer.

En pratique, nous l'utilisons également systématiquement en Master 2, avec des résultats très encourageants, pour la formation des étudiants aux techniques de simulation multi-agents, en particulier pour les initier à la simulation orientée interactions. Chaque rentrée universitaire est donc de fait l'occasion un peu forcée de se livrer à une maintenance qui intègre les retours des étudiants et les sujets des projets réalisés, ainsi que des mises à jour consécutives à celles de NetLogo et l'élimination de *bugs* résiduels.

Enfin, l'équipe de NetLogo nous a fait l'amitié de référencer notre extension directement à partir du site web de la plateforme, ce qui permet d'assurer une visibilité plus grande (les deux tiers des visiteurs proviennent de cette source, d'après Google Analytics).

☞ `http://ccl.northwestern.edu/netlogo/resources.shtml`, rubrique « *Tools* »

Deuxième partie

Travaux en cours et perspectives

A change of work is the best rest.

Doyle [94]

Chapitre 4

La simulation multi-niveaux

> Lorsque nous y serons parvenus par l'art du discours ou du calcul, [...] philosophant le long des degrés de l'échelle, c'est-à-dire de la nature, pénétrant toutes choses depuis le centre jusqu'au centre, alors nous pourrons tantôt descendre en démembrant avec une force titanesque l'un dans le multiple, [...] tantôt monter en rassemblant avec une force apollinienne le multiple dans l'un.
>
> Pic de la Mirandole [159]

Nous avons montré au début de ce mémoire que la simulation multi-agents, depuis quelques années, était utilisée pour modéliser, expliquer et prédire le fonctionnement de systèmes complexes fonctionnant sur une large échelle : c'est cette évolution qui nous a poussés à développer l'approche orientée interactions (IODA).

Toutefois, l'examen des problématiques auxquelles nous sommes confrontés de façon récurrente suggère que la question du *large échelle* se subdivise en plusieurs sous-problèmes distincts, certains appelant des réponses logicielles, d'autres une modélisation adaptée, tels que :

- des architectures logicielles permettant le traitement (en termes de temps d'exécution, de mémoire, de distribution physique, etc.) d'un nombre d'agents très élevé (de l'ordre de 10^5 – 10^6) ;
- la représentation de niveaux d'observation plus ou moins fins ;
- le choix de modéliser un sous-système par un seul agent ou par une organisation d'agents ;
- la représentation des comportements d'une entité selon qu'elle opère comme membre d'une organisation ou comme extérieure à cette organisation ;
- le regroupement de micro-agents en un macro-agent capable soit d'agréger l'ensemble de leurs comportements, soit de réifier une abstraction structurelle ou organisationnelle.

Ces problématiques ont conduit depuis quelques années seulement à une reformulation des infléchissements à donner au domaine de la simulation multi-agents. La première de ces nouvelles orientations, que nous allons aborder dans ce chapitre, consiste à tenter de décomposer les simulations en tenant compte de l'existence de divers niveaux d'organisation ou d'abstraction. Dans le chapitre 5, nous étudierons la possibilité de paramétrer ou même de construire des comportements à partir des masses de données issues de l'observation des systèmes réels. Enfin, nous dirons également quelques mots dans le chapitre 6 de la façon dont la question des architectures logicielles s'articule par rapport aux autres.

4.1 Contexte scientifique

4.1.1 L'émergence de la simulation multi-agents multi-niveaux

Les problèmes que soulèvent les systèmes complexes large échelle ne viennent pas tant de l'*échelle* de la simulation (i.e. la quantité d'agents ou de familles d'agents) que de la *coexistence de différents niveaux d'organisation ou d'analyse, ou de différentes échelles*. Par exemple, la biologie manipule des entités allant de l'organisme à la molécule, en passant par les tissus et les cellules [56]. Les physiciens, quant à eux, ont depuis longtemps développé leurs propres méthodes de modélisation multi-échelles à partir de systèmes équationnels [107, 104]. On voit également se développer des simulations multi-modèles [69, 86] qui cherchent à coupler des paradigmes différents (équations, réseaux bayésiens, automates cellulaires, systèmes multi-agents) selon le niveau d'observation à représenter. Ces approches partagent un inconvénient majeur : le manque d'*intelligibilité* des modèles pour les thématiciens, et leur faible *révisabilité*. En outre, le couplage de modèles de natures différentes accroît le risque d'introduire des biais dans les résultats de simulation.

La simulation multi-niveaux, d'après Morvan [163], pose au moins trois grandes questions théoriques :

— l'établissement de méta-modèles génériques et de moteurs de simulations ;
— la détection et la réification de phénomènes émergents ;
— la définition de représentations génériques pour les entités agrégées.

Un objectif à long terme : agentifier les structures émergentes

Ces deux derniers points présentent un intérêt certain et les problématiques qu'ils soulèvent sont particulièrement stimulantes. Les travaux précurseurs dans ce domaine sont ceux de Servat [195] autour de la réification de structures hydrologiques (projet RIVAGE, IRD). Les agents « boules d'eau » s'agrègent automatiquement pour former des entités macroscopiques hydrologiques (mares, ravines, etc.). Les agents correspondants sont créés dynamiquement pour remplacer les boules d'eau quand celles-ci sont entourées d'agents similaires. Inversement, le macro-agent peut se dissoudre en micro-agents lorsque des conditions d'extinction sont remplies (par exemple débordement d'une mare). Néanmoins, l'automatisation des comportements et de la création/destruction de macro-agents est largement simplifiée par le caractère physique et indifférencié des boules d'eau.

Depuis, d'autres travaux ont été menés pour identifier, voire réifier des structures émergentes [102, 138, 87, 78] mais la difficulté majeure reste l'agrégation automatique de comportements pertinents. Toutefois, si l'on souhaite construire une approche de simulation multi-niveaux générique, il semble plus judicieux (quoique peut-être plus austère) de commencer par définir un cadre général pour cela, à la fois en termes de forme des modèles conceptuels associés et d'algorithmes de simulation.

Un cadre générique pour le multi-niveaux

Notre objectif à plus court terme, dans ces travaux commencés courant 2010 [9, 26], est donc plutôt de construire une méthode de simulation multi-niveaux « homogène » en terme de paradigme, c'est-à-dire dans laquelle on puisse, du niveau macroscopique au niveau microscopique, ne manipuler que des *agents* et leurs *interactions* au sein *d'environnements* correspondant à des niveaux d'analyse différents. Cette homogénéité de paradigme n'a de pertinence, à notre sens, que si l'on opte pour une **intégration forte** des niveaux du modèle, impliquant un *partage des agents entre les niveaux* [163].

Il faut pour cela résoudre deux problèmes relativement indépendants :

1. **La question de « l'emboîtement »** : si les systèmes complexes sont souvent composés de sous-systèmes, ces derniers peuvent intervenir dans plusieurs structures macroscopiques, de sorte qu'une décomposition arborescente peut s'avérer très vite restrictive. La première étape consiste donc à formaliser les *relations qui peuvent être nécessaires entre agents et environnements* pour représenter dans un paradigme homogène la diversité des rapports entre systèmes et sous-systèmes, et ce d'une façon *générique* (indépendante de tout cadre applicatif) et *dynamique* (ces relations peuvent évoluer au cours de la simulation).
2. **La question de la dépendance des comportements au contexte** : le comportement d'un agent, dans un système complexe, dépend en général du sous-système dans lequel il opère. La séparation déclaratif/procédural introduite par IODA nous sera fort utile pour cela.

Bien entendu, un tel cadre de modélisation n'a d'intérêt que s'il conduit à un modèle computationnel et à une implémentation. C'est bien le cas de notre proposition, PADAWAN (pour *Pattern for an Accurate Design of Agent Worlds in Agent Nests*) que nous décrivons ci-après (§4.2). Nous verrons aussi que cette approche initialement conçue pour la simulation trouve également des applications à travers des méthodes de résolution distribuée de contraintes en cartographie (cf. §4.4). Enfin, nous reviendrons dans le chap. 6 sur les possibilités d'extension de notre approche par des méthodes d'identification et de réification de structures émergentes.

4.1.2 Travaux connexes

Définition de structures emboîtées

L'idée de décomposer un système en sous-systèmes n'est pas neuve dans le cadre des SMA. La plateforme SWARM [158] est une des premières qui ait explicitement introduit des échelles multiples dans les simulations centrées individus, à travers une organisation récursive en *swarms* contenant des agents et leur ordonnanceur, les *agents*

pouvant être eux-mêmes des *swarms*. C'est toutefois un modèle strictement arborescent, figé, où le comportement d'un *swarm* est réduit au comportement collectif des agents qui le constituent.

D'autres plateformes, comme GEAMAS [148] ou GAMA [203], permettent également de représenter des agrégations d'agents, voire de les détecter automatiquement ; toutefois elles reposent comme SWARM sur l'imbrication d'agents les uns dans les autres et ne permettent encore qu'une représentation arborescente, ce qui est plutôt limitatif. Il en va de même pour des architectures comme les systèmes holoniques [118]. D'autres approches permettent d'éviter une structure arborescente, mais elles ont souvent plutôt comme objectif une distribution physique des tâches [187].

Les méthodologies fondées sur une approche organisationnelle des SMA ont également proposé un cadre théorique et pratique pour la décomposition en sous-systèmes, comme AGRE (Agents, Groupes, Rôles + Environnement) [111] qui permet à un agent d'être à la fois dans l'environnement physique et de prendre part à environnements sociaux (groupes et organisations). Cette approche peut être adaptée à la simulation, par exemple en utilisant les groupes pour représenter des niveaux d'organisation dans un écosystème [54]. Néanmoins cette dichotomie physique/social nous semble restrictive dans la mesure où elle instaure une division arbitraire entre deux (et seulement deux) catégories d'environnements, tout en imposant l'unicité de l'environnement physique. Or l'environnement physique peut impliquer des milieux distincts où les règles pertinentes changent (par exemple, aquative *vs.* terrestre), et dans ce cas au moins il est sans doute plus simple de le découper en compartiments relativement indépendants.

Spécification locale de comportements

Il ne suffit pas de savoir décomposer un système : il faut encore pouvoir exprimer le fait que les comportements des agents sont dépendants du sous-système dans lequel l'agent est placé. Un formalisme intéressant nous est donné par les P-systèmes [171], un modèle de calcul Turing-complet inspiré de la biologie, qui repose sur les éléments suivants :

— des *membranes* contenant des symboles et des règles, et emboîtées de façons arborescente ;
— des *symboles* contenus dans les membranes sous forme de sacs (ou multiensembles), éventuellement étiquetés pour leur permettre de franchir une membrane ; un symbole spécial (δ) permet de dissoudre la membrane qui le contient ;
— des *règles* décrivant la transformation d'un sac de symboles en un autre (par exemple : $aab \rightarrow ac$) ; les règles placées dans une membrane expriment la façon dont les symboles qui y sont présents peuvent être transformés.

Le calcul, non-déterministe, consiste à appliquer itérativement le plus grand nombre possible de règles dont tous les symboles sont présents. Cette méthode se prête difficilement à la modélisation d'entités très complexes. Toutefois, on notera que les symboles n'ont aucun comportement en tant que tel, seul *l'ensemble des règles présentes dans la membrane où se trouvent ces symboles permet de définir leur fonction*. Autrement dit, on a d'une part *séparation entre entités et comportements*, et d'autre part *définition locale des comportements*. Ce principe est très proche de ce que nous avons fait dans IODA et souhaitons obtenir en modélisation multi-niveaux.

4.1.3 Vers un cadre multi-niveaux fortement intégré

Il apparaît dans les travaux cités ci-dessus que, la plupart du temps, l'accent est mis sur les problématiques de structuration des entités (ou d'emboîtement) du niveau microscopique ou atomique jusqu'au plus haut niveau de composition, en laissant au second plan la question de la modification des comportements en fonction du niveau où l'agent opère. C'est d'ailleurs sans doute le principal obstacle pour construire des cadres généraux permettant une intégration forte des différents niveaux d'observation. La seule autre approche à notre connaissance permettant d'aller dans ce sens est IRM4MLS [164], quasiment contemporaine de PADAWAN, et qui, étant basée sur IRM4S [154], peut exprimer de façon très claire les influences émises et les réactions calculées à chaque niveau. Néanmoins cette approche utilise un modèle à temps discret synchrone qui soulève des difficultés pour spécifier la fonction de réaction.

Il semble donc bien que ce qui pose problème dans le multi-niveaux, c'est la notion même de *niveau*... d'ailleurs Morvan [163] recense également les dénominations « multi-échelles » (*multi-scale*), « multi-couches » (*multi-layer*), « multi-perspectives », « multi-interactions », « multi-points de vue » (*multi-view*), etc. : autant dire que le concept est assez mal défini. Le principe de parcimonie réclame sa tête, mais nous allons surseoir à l'exécution.

En fait, il y a déjà assez de concepts utiles dans un SMA, pour se dispenser temporairement de celui de niveau. En effet, c'est le fait pour un agent *d'être situé* dans son environnement (autrement dit d'avoir une perception limitée, des relations de voisinage avec d'autres agents, la possibilité ou non de se mouvoir, etc.), qui rend son comportement adaptable aux changements et permet de construire des solutions cohérentes et éventuellement orientées vers des buts, explicites ou non. *L'environnement est le cadre spatial ou social fondamental qui structure les comportements d'un agent*. Par conséquent, la première étape à envisager pour introduire plusieurs niveaux, est de *s'appuyer sur la notion d'environnement*. Bien entendu, l'environnement doit également être associé à une *échelle d'observation* qui permet de caractériser la granularité temporelle et spatiale des phénomènes qui s'y déroulent, et de déterminer les agents qui ont vocation à y opérer.

Pour permettre une adaptation locale des comportements des agents au « niveau », et pour autoriser l'appartenance des agents aux divers « niveaux », la simulation « multi-niveaux » doit avant tout être une simulation « multi-environnements ». Cette condition nécessaire n'est pas suffisante et nous proposons de lui adjoindre les caractéristiques ci-dessous, dont nous montrerons comment les réaliser dans la section suivante :

- **Appartenance multiple** : toute entité doit pouvoir appartenir simultanément à plusieurs environnements ;
- **Dynamicité** : un agent doit pouvoir changer de niveau (d'environnement) à volonté si cela fait sens dans le modèle ; de plus il ne doit pas y avoir de limitation de principe à la création ou à la dissolution d'un niveau si cela est pertinent ;
- **Localité** : le comportement d'un agent doit être le résultat de sa présence dans un environnement donné (l'espace qui définit ses conditions d'existence) ;
- **Homogénéité** : dans IODA toute entité est un agent et tout comportement, une interaction ; de même il faut que la décomposition du monde ne s'appuie que sur des environnements.

4.2 Un modèle homogène pour les simulations multi-niveaux (PADAWAN)

Le modèle PADAWAN, que nous avons construit pour répondre aux critères ci-dessus, repose sur deux principes (nous renvoyons aux publications correspondantes pour plus de détails [9, 26]) :

— autoriser un agent à *être situé dans plusieurs environnements* simultanément, sans préjuger de ce que représentent ces environnements (monde physique, groupes, organisations, mémoire spatiale ou sociale, etc.), façon de généraliser les approches de type AGRE ;

— permettre *l'encapsulation dynamique* d'un environnement par un agent, donc construire les bases d'une structure récursive relativement classique (SWARM, P-systèmes), mais non figée.

Il s'appuie donc sur deux relations entre agents et environnements : la **situation** et l'**encapsulation**.

Dans la suite, nous désignerons par $\mathcal{E}$ l'ensemble des environnements utilisés dans une simulation, et par $\mathcal{A}$ l'ensemble des agents. Ces environnements doivent être vus comme des *espaces* de topologie quelconque, de dimensions quelconques. Nous allons montrer également comment les doter en outre d'une échelle temporelle propre et de règles comportementales différenciées qui s'appliquent aux agents qu'ils contiennent.

4.2.1 Relations entre agents et environnements

Relation de situation

Nos agents sont *situés*, donc ils appartiennent à au moins un environnement, dans lequel ils peuvent percevoir et agir. Nous autorisons un agent à être situé dans plusieurs environnements, ce qui nous amène à définir la **relation de situation d'un agent dans un environnement** :

Définition VII — Relation de situation.
Un agent $a \in \mathcal{A}$ **est situé** dans un environnement $e \in \mathcal{E}$, noté : $\mathbf{a} \lhd \mathbf{e}$, lorsque a peut percevoir, être perçu, agir ou subir des actions, dans e.

On note dans la suite respectivement le **lieu** de a (l'ensemble des environnements où a est situé) et le **contenu** de e (l'ensemble des agents situés dans e) :

$$\forall a \in \mathcal{A},\ \text{location}(a) \stackrel{\text{def}}{=} \{e \in \mathcal{E} \mid a \lhd e\} \quad \text{et} \quad \forall e \in \mathcal{E},\ \text{content}(e) \stackrel{\text{def}}{=} \{a \in \mathcal{A} \mid a \lhd e\}$$

On se donne également le droit d'écrire par exemple $\text{location}(a) \leftarrow \{e_1, e_2\}$ pour exprimer le fait qu'on place a exactement dans les environnements e_1 et e_2 (i.e. qu'on établit une relation de situation entre a et e_1 et une autre entre a et e_2). Le fait de travailler avec des agents situés impose : $a \in \mathcal{A} \mid \text{location}(a) = \emptyset$.

On appelle **agent simplement situé** tout agent $a \in \mathcal{A}$ situé dans un environnement unique. L'ensemble des agents simplement situés est noté $\mathcal{S}$:

$$\mathcal{S} \stackrel{\text{def}}{=} \{a \in \mathcal{A} \mid \exists e \in \mathcal{E}, \text{location}(a) = \{e\}\}$$

De même, on appelle **agent multi-situé** tout agent $a \in \mathcal{A}$ situé dans plusieurs environnements simultanément.

Les agents simplement situés et les agents-muti-situés sont bien entendu *caractérisés dynamiquement* : il n'y a donc pas de typage préalable et un agent peut changer de statut au cours de la simulation.

Un agent multi-situé peut représenter des situations très diverses : par exemple, il peut s'agir d'un agent jouant un rôle de frontière entre deux environnements physiques (une porte), qui doit donc percevoir ou être perçu par les autres agents de ces environnements. On peut aussi s'en servir pour représenter l'appartenance sociale d'un agent ou sa présence dans des espaces appartenant à des échelles différentes (par exemple une micropipette qui est à la fois à l'intérieur d'une cellule et dans la main du biologiste).

Relation d'encapsulation

La seconde relation du modèle représente la capacité, pour *a priori* n'importe quel agent, d'encapsuler un environnement (dans lequel vont pouvoir se situer d'autres agents). Comme nous l'avons dit, il est conceptuellement important de maintenir la distinction entre agent (entité) et environnement (espace), ne serait-ce que parce que les relations métriques ou topologiques « à l'intérieur » d'un agent sont bien des propriétés de l'*espace* et non de l'agent lui-même ; mais jusqu'à un certain point on peut considérer que ces agents « jouent le rôle » d'un environnement.

Définition VIII — Relation d'encapsulation.
Un agent $a \in \mathcal{A}$ **encapsule** un environnement $e \in \mathcal{E}$, noté : $\mathbf{a \sim e}$, lorsque a « contient » e. Un agent ne peut encapsuler qu'un seul environnement à la fois, et un environnement n'être encapsulé que par un seul agent.

Par construction, on pose qu'il existe un environnement unique e^0 non encapsulé par un agent (e^0 correspond à « l'Environnement » unique d'un SMA classique). Cet environnement est appelé « environnement racine » (ou « environnement zéro ») de la simulation. On note $\mathcal{E}^\star \stackrel{\text{def}}{=} \mathcal{E} \setminus \{e^0\}$ l'ensemble des environnements encapsulés par des agents.

On appelle **agent régulier** tout agent n'encapsulant pas d'environnement, et **agent-compartiment** tout agent encapsulant un environnement. L'ensemble des agents-compartiments est noté $\mathcal{C}$: $\mathcal{C} \stackrel{\text{def}}{=} \{a \in \mathcal{A} \mid \exists e \in \mathcal{E}^\star, a \sim e\}$ L'ensemble des agents réguliers est noté $\mathcal{R}$: $\mathcal{R} \stackrel{\text{def}}{=} \mathcal{A} \setminus \mathcal{C}$

Par ailleurs on définit respectivement **l'hôte** de e comme l'unique agent qui encapsule e, et **l'espace** de a comme l'unique environnement encapsulé par a :

$$\forall e \in \mathcal{E}^\star,\ \text{host}(e) \stackrel{\text{def}}{=} !a \in \mathcal{C} \mid a \sim e \quad \text{et} \quad \forall a \in \mathcal{C},\ \text{space}(a) \stackrel{\text{def}}{=} !e \in \mathcal{E}^\star \mid a \sim e$$

Utilisation conjointe des deux relations

En combinant la situation dans un environnement et l'encapsulation d'un environnement par un agent, on peut définir de nouveaux concepts et les bases d'une architecture multi-niveaux.

En particulier, on peut caractériser la structure de la simulation au moyen de nouvelles relations : l'inclusion et l'hébergement.

Définition IX — Inclusion d'environnements.
Un environnement e_1 est **inclus** dans un environnement e_2 (noté $e_1 \subset e_2$) si et seulement si :

$$\exists a \in \mathcal{C} \mid a \sim e_1 \wedge a \lhd e_2$$

Définition X — Hébergement d'agents.
Un agent a_1 est **hébergé** par un agent a_2 (noté $a_1 \sqsubset a_2$) si et seulement si :

$$\exists e \in \mathcal{E}^\star \mid a_1 \lhd e \wedge a_2 \sim e$$

Ces deux relations permettent de construire le **graphe d'inclusion** (ou graphe d'environnements) d'une simulation, graphe orienté défini par l'ensemble des relations d'inclusion ($\subset$) entre environnements. Ce graphe est dynamique puisqu'il représente les relations effectives de situation et d'encapsulation à chaque instant dans la simulation. De même, le **graphe d'hébergement** (ou graphe d'agents) d'une simulation est le graphe orienté défini par l'ensemble des relations d'hébergement ($\sqsubset$) entre agents. On peut également construire un **graphe mixte** en identifiant les environnements encapsulés à leur compartiment (cf. fig. 4.1 ci-dessous, ainsi que 4.4 dans la section 4.3).
Exemple. — Soient les environnements e_1 et e_2, les agents $\{a_i\}_{i=1\ldots6}$ tels que : $a_1 \sim e_1, a_2 \sim e_2, \text{location}(a_1) = \text{location}(a_2) = \{e^0\}, \text{location}(a_3) = \{e_1\}, \text{location}(a_4) = \{e_1, e_2\}, \text{location}(a_5) = \{e_2\}, \text{location}(a_6) = \{e^0\}$. La figure 4.1 présente les graphes d'inclusion (4.1a) et d'hébergement (4.1b), ainsi que le graphe mixte (4.1c) correspondant à cette situation. En pratique, les agents qui peuvent interagir sont : a_1, a_2 et a_6 (dans e^0), a_3 et a_4 (dans e_1), a_4 et a_5 (dans e_2).

Par extension, on note $e_1 \subseteq e_2$ (inclusion transitive) si et seulement si : $e_1 = e_2$ ou $e_1 \subset e_2$ ou $\exists e \in \mathcal{E} \mid e_1 \subseteq e \wedge e \subseteq e_2$ (i.e. s'il existe un chemin entre e_1 et e_2 dans le graphe d'inclusion) ; de même $a_1 \sqsubseteq a_2$ (hébergement transitif) si et seulement si : $a_1 = a_2$ ou $a_1 \sqsubset a_2$ ou $\exists a \in \mathcal{A} \mid a_1 \sqsubseteq a \wedge a \sqsubseteq a_2$.

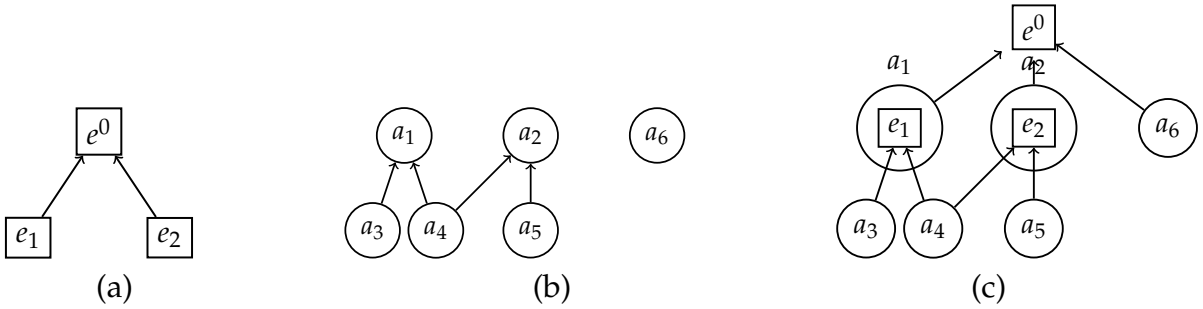


FIGURE 4.1 – (a) Graphes d'inclusion (resp. (b) d'hébergement, et (c) mixte) correspondant à l'exemple décrit dans le texte (p. 76). Par convention les agents sont représentés par des cercles, les environnements par des carrés. La superposition cercle/carré (c) correspond à l'encapsulation d'un environnement par un agent ($\sim$). Les arcs représentent respectivement la relation $\subset$ (a), la relation $\sqsubset$ (b), et la relation $\lhd$ (c).

Pour éviter des situations paradoxales (comme deux agents qui s'hébergent mutuellement), on impose dans toute la suite qu'à tout moment le graphe d'inclusion de la simulation *ne comporte pas de cycles*. Cela revient à dire que la relation $\subseteq$ est une **relation d'ordre** (le graphe d'inclusion est alors un demi-treillis de borne supérieure e^0).

Dès lors, la notion quelque peu qualitative de **niveau** d'un environnement (ou du compartiment associé) peut alors être définie formellement comme la *distance minimale à* e^0, ce qui permet de caractériser une partie de l'organisation d'un système et de ses sous-systèmes. On voit par là que dans PADAWAN, le concept de niveau n'est pas premier : c'est l'environnement qui prime, le niveau n'en est qu'une propriété.

Comme nous allons le voir, l'encapsulation d'un environnement quelconque (c'est-à-dire un espace) dans un agent doté de ses propres comportements, permet une grande variété d'usages, dans la mesure où chaque environnement peut modéliser aussi bien un espace physique qu'un réseau d'accointances ou toute forme d'organisation. On peut aussi bien les utiliser pour représenter des points de vue différents sur le système, que pour rendre compte d'un « emboîtement » de structures et de sous-structures, où « zoomer » d'un système à ses composants revient à passer d'un agent compartiment vu de façon atomique aux agents qu'il héberge.

À l'aide des relations ainsi construites, il est possible non seulement de définir la structure initiale d'une simulation et son évolution, mais également de donner aux agents les briques comportementales (primitives) nécessaires pour manipuler cette structure de façon cohérente. Nous allons maintenant détailler ce point.

4.2.2 Dépendance des comportements au contexte : force de IODA

La séparation que l'approche IODA introduit entre entités (agents) et comportements (interactions) est particulièrement utile lorsque l'on veut pouvoir spécifier des comportements différents selon le niveau où agit un agent. En effet, *il suffit essentiellement d'associer une matrice d'interaction* M_i *à chaque environnement* e_i. À ce principe général s'ajoutent plusieurs mécanismes que nous expliquons ci-dessous.

Notion de face d'un agent

Dans le cas général d'un agent a tel que location$(a) = \{e_1, ..., e_n\}$, nous procédons en une décomposition de l'agent a en une partie « centrale » non située, et en un ensemble de parties « périphériques » situées dans chaque environnement e_i. Nous appelons **« noyau de a »** la partie non située (notée $a_{|\bullet}$) et **« face de a dans $\mathbf{e_i}$ »** (notée $a_{|e_i}$) la partie de a située dans e_i (soit en fait a vu comme agent du seul environnement e_i). À $a_{|e_i}$ sont associées des propriétés spécifiques à sa situation, comme une position dans e_i ainsi qu'un halo de perception qui permet de calculer les voisins de a *dans* e_i, i.e. les agents de e_i avec lesquels a peut tenter d'interagir selon la matrice d'interaction M_i de e_i. Enfin, pour agir en tant qu'*hôte* vis-à-vis des agents qu'il héberge, un agent compartiment doit également disposer d'une face située dans l'environnement qu'il encapsule — avec cette caractéristique particulière que sa distance à n'importe lequel des agents hébergés est nulle (il les perçoit tous et est à portée d'action de tous). Cette « **face-hôte** » (*hostface*) est notée $a_{|\square}$.

Ainsi, on peut voir un agent issu du modèle multi-niveaux comme la superposition d'un agent « désincarné » qui ne perçoit ni n'agit (il se limite à un ensemble d'état et à des fonctions de calcul, de décision) et d'un ou plusieurs organes sensori-moteurs plus ou moins élaborés qui chacun, pris individuellement dans l'environnement où ils sont situés, peuvent être traités tout à fait comme des agents d'une simulation ordinaire. Le pseudo-agent $a_{|e_i}$ (ou $a_{|\square}$) est simplement soumis au principe de sélection d'interaction classique de IODA relativement à l'environnement e_i et à la matrice d'interaction M_i, la seule différence avec une approche « plate » (dans un environnement unique) étant que les primitives exécutées par $a_{|e_i}$ ont accès si nécessaire à l'état et aux fonctions du pseudo-agent noyau $a_{|\bullet}$.

Spécificité des matrices d'interaction liée à l'encapsulation

Par rapport à l'approche IODA « classique », il faut également pouvoir exprimer des interactions « verticales », i.e. entre les agents compartiments et ceux qu'ils hébergent. Pour cela, il suffit d'ajouter une ligne et une colonne dans chaque matrice d'interaction.

Interactions hébergeur-hébergé. En général, un agent compartiment doit pouvoir interagir avec les agents qu'il héberge. Pour réaliser cela, il suffit d'ajouter dans la matrice d'interaction une **ligne générique « hôte »** : elle sert à spécifier quelles interactions le compartiment peut effectuer avec l'environnement qu'il encapsule et avec les agents qu'il héberge (voir l'exemple §4.3).

Interaction hébergé-hébergeur. Réciproquement, un agent doit aussi pouvoir interagir avec l'agent compartiment qui l'héberge, ne serait-ce que pour pouvoir en sortir. Il suffit ici d'ajouter à la matrice d'interaction une **colonne générique « hôte »** qui spécifie quelles interactions le compartiment peut subir de la part des agents qu'il héberge.

Grâce à ces deux extensions, l'uniformité de IODA est respectée car seule la matrice d'interaction est modifiée, sans aucun changement dans les algorithmes de sélection d'interaction, comme nous allons l'expliquer ci-dessous.

Gestion des échelles de temps et ordonnancement de la simulation

La simulation multi-niveaux demande de gérer non seulement diverses échelles spatiales, mais aussi des échelles de temps différentes. Dans la plupart des plate-formes qui offrent cette possibilité, deux méthodes sont principalement utilisées. La première s'appuie sur le formalisme à événements discrets DEVS [219], déjà mentionné à propos du moteur IODA (§2.2.3). La seconde consiste à doter les macro-agents (ceux qui « contiennent » les autres) de leur propre ordonnanceur (SWARM, GAMA).

Notre approche en la matière ne vise pas à proposer une solution universelle, mais il nous semble plus élégant et cohérent de considérer que le temps est une propriété liée à *l'espace* donc à *l'environnement*, et non aux directement aux agents. Notre propos ici est seulement de montrer comment doter les environnements de caractéristiques temporelles propres et quelles en sont les conséquences pour l'ordonnanceur, aussi avons-nous choisi de conserver des simulations à temps discret à approche séquentielle, avec un pas de temps δt. Néanmoins, notre approche est compatible avec des modèles plus sophistiqués, comme le modèle à temporalité de Payet *et al.* [172].

Dans ce cadre, chaque environnement est associé à sa propre échelle de temps, au moyen de la fonction suivante :

$$\begin{array}{rcl} \chi : \mathcal{E} & \to & \mathbb{N}^* \times \mathbb{N} \\ e & \mapsto & (N, \varphi) \end{array}$$

où N et φ représentent la **période** et la **phase** attachées à l'environnement e.

La période permet d'indiquer qu'il « se passe quelque chose » dans e tous les $N\delta t$; la phase, que e est « décalé » par rapport au début de la simulation d'un temps $\varphi\delta t$. Rien n'empêche que cette assignation change au cours de la simulation (par exemple pour tenir compte des variations d'activité d'une cellule en fonction de la température) : dans ce cas χ est alors aussi fonction de t.

On peut alors dire qu'un environnement e est **activé au temps** t lorsque : $t \equiv \varphi \pmod N$. On désigne par $\mathcal{E}_{\text{act}}(t)$ l'ensemble des environnements activés au temps t.

Les agents susceptibles d'agir au temps t (i.e. agents actifs susceptibles d'effectuer une interaction, ou labiles susceptibles de changer d'état, cf. §2.2.1) sont donc ceux situés dans les environnements activés au temps t. En pratique, comme chaque agent agit selon la matrice d'interaction de l'environnement où il se trouve, il faut recenser les faces situées dans des environnements activés, soit les couples $(a, e) \in \mathcal{A} \times \mathcal{E}$ activés à l'instant t : on les désigne par $\mathcal{A}_{\text{act}}(t)$.

$$\mathcal{A}_{\text{act}}(t) \stackrel{\text{def}}{=} \bigcup_{e \in \mathcal{E}_{\text{act}}(t)} \left(\bigcup_{a \in \text{content}(e)} \{(a, e)\} \right)$$

L'algorithme que nous utilisons pour l'ordonnancement des agents est une version simplifiée de l'ordonnancement conservatif de HLA [114], dans la mesure où chaque environnement peut être vu comme une simulation basée sur des pas de temps ; il est évidemment capital de prévoir un ordonnancement paramétrable des interactions.

Pour la sélection d'interaction au temps t, les couples de $\mathcal{A}_{\text{act}}(t)$ sont donc soumis préalablement à une **politique d'ordonnancement** (par défaut : mélange aléatoire)

qui va déterminer l'ordre dans lequel les faces correspondantes évaluent et choisissent les interactions à réaliser. On peut envisager diverses politiques d'ordonnancement, comme un tri portant sur les agents, ou sur les environnements, ou sur les deux. Entre autres, on peut mentionner un tri portant sur les environnements selon leur niveau (tri ascendant ou descendant) : ainsi, faire agir les niveaux les plus élevés en premier pourrait donner une approche « émergentiste » ; ou encore, un tri portant sur les environnements selon leur période (tri ascendant ou descendant). Selon toute vraisemblance choisir une politique d'ordonnancement pertinente dépend du domaine d'application.

Une fois établi l'ordre de $\mathcal{A}_{\text{act}}(t)$, il reste à opérer une sélection d'interaction classique (à la IODA) sur les *faces* correspondantes, i.e. pour chaque couple (a,e) la face $a_{|e}$ est susceptible d'interagir comme source au plus une fois. Quant au noyau $a_{|\bullet}$, il n'effectue ni ne subit par lui-même aucune interaction : seule l'activité de ses faces, *via* leurs primitives, est susceptible de l'affecter.

Ajout de primitives spécifiques

L'introduction des relations de situation et d'encapsulation dans l'approche orientée interactions va de pair avec l'ajout de primitives minimales et génériques permettant aux agents de construire des interactions relatives à la manipulation de ce système multi-niveaux. Nous allons en donner quelques exemples, en montrant que leur sémantique peut être définie de façon univoque à partir des notions définies plus haut. Bien évidemment des primitives supplémentaires peuvent être développées selon les besoins particuliers de chaque domaine d'application.

Primitives de test. En premier lieu, les agents disposent de fonctions booléennes permettant de les caractériser : agent régulier *vs.* compartiment, agent simplement situé *vs.* multi-situé, etc. Un agent peut également tester s'il est hébergé par un autre agent. Cela ne présente guère de difficulté.

Primitives d'action. Définir la sémantique des actions réalisables demande un peu de rigueur, notamment parce qu'il faut s'assurer que la définition vaut aussi bien pour le cas des agents multi-situés ou des agents compartiments que pour celui des agents simplement situés ou des agents réguliers. De plus, les primitives doivent garantir à tout moment qu'aucun cycle n'apparaît dans le graphe d'inclusion.

Ainsi, par exemple, la primitive « entrer dans un compartiment » `enter(a,c)` n'a de sens que si a et c sont situés dans au moins un environnement commun et que $c \not\sqsubseteq a$: dès lors, il faut modifier le lieu de a en remplaçant tous ses environnements communs avec c par l'environnement encapsulé par c : $\text{location}(a) \leftarrow (\text{location}(a) \setminus \text{location}(c)) \cup \{\text{space}(c)\}$

Dans certains cas, la primitive peut être paramétrée par la donnée d'une fonction. Par exemple, la division d'un compartiment c suppose de définir une **politique de déplacement** π pour savoir lesquels des agents qui étaient hébergés par ce compartiment doivent en changer et se placer dans le nouveau. Pour exécuter `divide(c)`, on doit donc créer un compartiment c' situé lui-même dans $\text{location}(c)$ puis définir la politique : $\pi : \text{content}(\text{space}(c)) \to \{\bot, \top\}$. Dès lors, pour tout agent $a \sqsubset c$ pour lequel

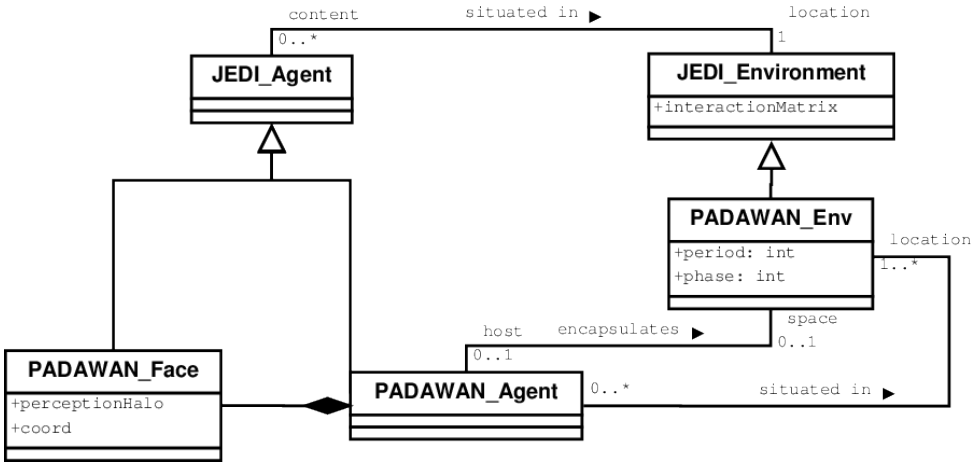


FIGURE 4.2 – Diagramme des classes montrant l'implémentation effective de PADAWAN au sein de la plateforme JEDI. Le modèle conceptuel manipule un agent « monolithique » (`PADAWAN_Agent`) qui est automatiquement décomposé dans la plateforme en un noyau non situé et une collection d'agents faces (`PADAWAN_Face`) situés chacun dans un environnement unique, bénéficiant ainsi du même fonctionnement que des agents JEDI classiques.

$\pi(a) = \top$, il faut remplacer dans location(a), space(c) par space(c'). La politique par défaut est une répartition aléatoire des agents hébergés.

Nous avons de même défini des primitives permettant de détruire (récursivement) un environnement ou un agent, de créer un agent dans des environnements spécifiés, d'entrer et de sortir d'un compartiment, de fusionner des compartiments, de dissoudre un compartiment (en situant les agents qu'il héberge dans les environnements où il était lui-même situé), de transformer un agent régulier en compartiment et vice-versa, de changer d'environnement en « traversant » un agent multisitué (porte), etc.

4.2.3 Implémentation au sein du moteur JEDI

En pratique, pour implémenter PADAWAN dans la plateforme « de référence » JEDI, nous avons réifié le concept de face des agents comme indiqué sur la figure 4.2. L'entité qui apparaît comme un « individu » (utilisée pour l'analyse et manipulée par l'utilisateur) dérive de la classe `PADAWAN_Agent`, mais toute mise en situation d'un agent dans un environnement crée en fait une face (`PADAWAN_Face`) qui se comporte comme un agent JEDI ordinaire dans son environnement unique.

L'ordonnanceur de JEDI a également été modifié pour calculer, à chaque pas de temps, les faces situées dans des environnements activables, puis leur appliquer une politique d'ordonnancement, et enfin leur faire réaliser la sélection d'interaction relativement à la matrice de leur environnement. Le choix de la politique d'ordonnancement (cf. § 4.2.2) est un paramètre du moteur JEDI (par défaut, ordre aléatoire).

Au cours d'un pas de temps, chaque face (en tant qu'agent JEDI « classique ») peut interagir au plus une fois comme source (au cours du tour de parole), et potentiellement plusieurs fois comme cible (i.e. choisi par d'autres agents sources). Un agent PADAWAN possédant des faces dans deux environnements activés est donc

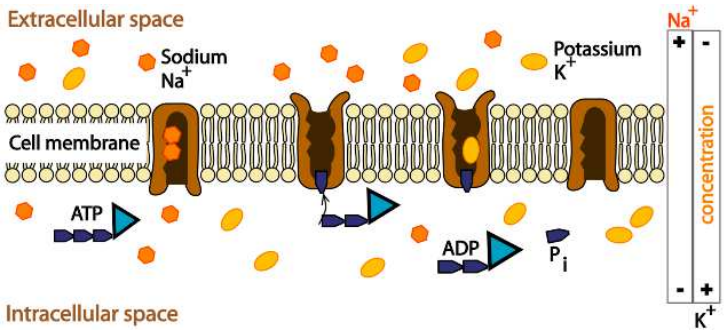


FIGURE 4.3 – Principe de la pompe Na^+/K^+ : la pompe est une protéine membranaire. Quand elle est ouverte vers l'intérieur de la cellule, elle peut capter 3 ions sodium Na^+. Une molécule d'ATP peut alors phosphoryler la pompe : sa conformation spatiale change et elle libère les ions sodium hors de la cellule. Dès lors, elle peut capter 2 ions potassium K^+, ce qui provoque sa déphosphorylation : elle reprend sa première conformation et libère les ions potassium à l'intérieur de la cellule. (Source de l'image : Wikimedia Commons)

susceptible d'être source de deux interactions (une dans chaque environnement). Il incombe donc *aux interactions* (i.e. au concepteur de la simulation) d'assurer le cas échéant la cohérence des actions effectuées par l'agent, via ses faces, dans les environnements où il se situe. Si par exemple une face exécute une primitive « grossir » et une autre « maigrir », cela veut dire que deux interactions, comportant chacune l'une de ces primitives, étaient réalisables simultanément : selon l'implémentation donnée aux primitives, cela peut être une erreur de conception, mais aussi la fusion de forces antagonistes issues d'environnements différents, ou encore des transformations qui n'affectent que les faces et pas le noyau de l'agent (apparence de l'agent dans chaque environnement par exemple).

4.3 Un exemple de modèle PADAWAN

Nous allons maintenant illustrer cette approche au moyen d'un court exemple qui vise surtout à expliquer comment les concepts décrits ci-dessus sont utilisés de façon concrète. Un autre exemple est donné dans [26]. Ici, nous modélisons la pompe sodium-potassium (qui régule le potentiel de repos des cellules vivantes), dont le fonctionnement SCHEINER-BOBIS [192] est décrit sur la figure 4.3. La modélisation commence par l'identification des familles d'agents nécessaires, de leur capacité à héberger ou non d'autres agents (autrement dit, à être des compartiments), et conduit au graphe mixte de la figure 4.4. Le modèle des comportements fait appel à trois matrices d'interactions (cf. tab. 4.1) qui suffisent à décrire de façon simple le comportement des agents impliqués en fonction de l'environnement dans lequel ils se trouvent (à l'intérieur de la cellule, à l'intérieur de la pompe ou à l'extérieur de la cellule). La pompe elle-même est un agent compartiment multi-situé qui appartient à la fois à l'intérieur et à l'extérieur de la cellule.

À l'intérieur de la cellule, les agents peuvent se déplacer aléatoirement (Errer) ; les ions sodium peuvent entrer dans la pompe (Entrer possède évidemment des conditions qui vérifient la conformation de la pompe). La molécule d'ATP a la capacité d'induire

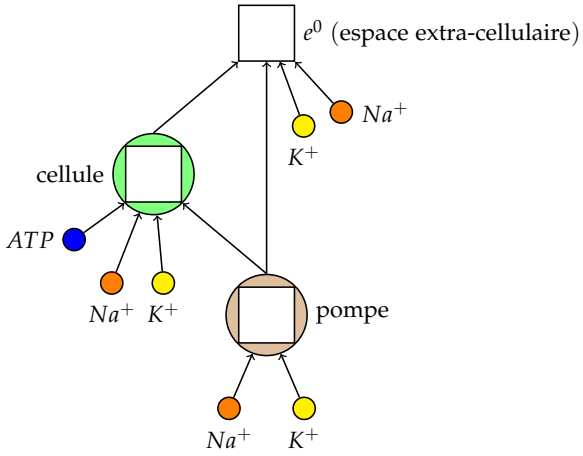


FIGURE 4.4 – Graphe mixte (environnements et agents) correspondant à la modélisation multi-niveaux de la pompe sodium-potassium. Par souci de simplification, un seul agent de chaque famille a été représenté dans chaque environnement (en pratique bien sûr il peut y avoir plusieurs ions sodium dans la cellule, dans la pompe ou à l'extérieur, ou même plusieurs cellules chacune dotée de plusieurs pompes, etc.)

Intérieur de la cellule		
SOURCES \ CIBLES	∅	Pompe
Na^+	Errer (0)	Entrer (1 ; 1)
K^+	Errer (0)	
ATP	Errer (0)	Phosphoryler (1 ; 1)

Extérieur de la cellule		
SOURCES \ CIBLES	∅	Pompe
Na^+	Errer (0)	
K^+	Errer (0)	Entrer (1 ; 1)

Intérieur de la pompe				
SOURCES \ CIBLES	∅	*hôte*	Na^+	K^+
hôte	Déphosphoryler (1)		Libérer (0 ; 0)	Libérer (0 ; 0)
Na^+	Errer (0)	Sortir (1 ; 0) SeLier (1 ; 0)		
K^+	Errer (0)	Sortir (1 ; 0) SeLier (1 ; 0)		

TABLE 4.1 – Les trois matrices d'interaction utilisées dans la simulation de la pompe sodium/potassium.

la phosphorylation de la pompe (qui change de conformation). À l'extérieur de la cellule, les agents peuvent se déplacer, et les ions potassium peuvent entrer dans la pompe. À l'intérieur de la pompe, les ions peuvent se déplacer ; ils peuvent également effectuer deux interactions avec leur hôte (si les conditions de ces interactions sont remplies, i.e. selon l'état de la pompe) : Sortir (si les ions sont libres) et SeLier sur un des sites disponibles (déterminés par l'état de la pompe). Enfin, la pompe elle-même, en tant qu'hôte, peut effectuer des interactions : se Déphosphoryler quand deux ions potassium sont fixés, et Libérer des ions selon sa conformation.

L'intérêt de cette décomposition en trois environnements est ici de *simplifier* considérablement la conception des comportements et de les décrire sous une forme *modulaire* : un modèle complexe, tel que la représentation de processus biochimiques, peut alors être construit par *composition* d'environnements et d'agents dont on sait, localement, spécifier les matrices d'interaction.

Cet exemple a été implémenté et testé dans JEDI. La configuration d'une expérience est un peu plus compliquée que pour une simulation « plate », car il faut définir l'état initial des relations d'encapsulation et de situation, d'où l'usage d'un fichier XML approprié.

4.4 Transposition à la résolution de contraintes pour la généralisation cartographique

Bien que PADAWAN ait été conçu à des fins de *simulation*, le contexte multi-niveaux où il se place, et le caractère très générique des relations introduites entre agents et environnements, permettent d'envisager de transposer ce formalisme à d'autres domaines d'utilisation des SMA. En l'occurrence, la présentation du modèle aux JFSMA en 2010 a donné lieu à la découverte de problématiques convergentes avec nos collègues Cécile Duchêne et Guillaume Touya du COGIT, à l'IGN [102].

La généralisation cartographique

L'utilisation de SMA en cartographie vise à automatiser un processus complexe et coûteux : la **généralisation cartographique**, qui consiste essentiellement, lors d'une diminution d'échelle (autrement dit lorsque l'on souhaite représenter une zone plus large sur la même surface), à réduire le niveau de détail dans les données géographiques d'une carte pour la rendre intelligible [60]. Pour cela, les cartographes doivent vérifier des contraintes portant sur les *symboles cartographiques*, et appliquer le cas échéant des actions correctives. Les routes, par exemple, doivent respecter une largeur minimale pour être visibles, sans rapport d'échelle avec leur largeur réelle. Il faut également assurer un espacement minimal entre objets pour pouvoir les séparer visuellement sans effort. De même, certaines informations pertinentes doivent être accentuées visuellement pour rester compréhensibles, quitte à supprimer certains détails, changer la forme et l'orientation des objets, en agréger, etc.

Les recherches menées au COGIT visent notamment à automatiser autant que possible ces opérations. Deux approches principales y ont été développées, mettant en œuvre un processus de résolution de contraintes :

— le modèle AGENT [139, 188], dans lequel les objets géographiques sont structurés de façon arborescente (par exemple des bâtiments contenus dans des îlots, eux-mêmes contenus dans des villes) : il permet à des entités représentant des groupes organisés d'objets cartographiques, appelés agents « méso », de gérer la résolution des conflits entre agents « micro » (par exemple les îlots peuvent éliminer ou déplacer certains de leurs bâtiments, et les villes transformer leurs îlots en un bloc compact) ;
— le modèle CartACom [100, 101], dans lequel les objets géographiques interagissent au même niveau avec leurs voisins pour résoudre des contraintes locales (espacement, alignement, etc.).

Ces modèles, en l'état actuel, ne sont pas utilisables simultanément. Une métaheuristique, CollaGen [208], permet de les appliquer alternativement pour optimiser la satisfaction des diverses contraintes en fonction du contexte.

Introduction d'une représentation multi-niveaux

En donnant aux objets cartographiques un cadre multi-niveaux approprié, on peut espérer la possibilité d'uniformiser au moins ces deux approches de résolution, en combinant des interactions verticales et horizontales, mais également de permettre des interactions « diagonales » (par exemple entre un bâtiment et un îlot adjacent dont il ne fait pas partie). De plus, l'utilisation du cadre conceptuel défini dans PADAWAN permet de traiter de façon uniforme des groupes correspondant à des « emboîtements » reflétant des contraintes et des situations très diverses : par exemple, un ensemble d'agents peut en utiliser un autre comme support spatial (des stations de bus le long d'une route), ou être dans les uns par rapport aux autres dans une relation de composition (groupes de bâtiments formant une structure particulière et pouvant être réifiée par un agent méso approprié) [7]. Enfin, la réification explicite des environnements manipulés (actuellement indissociables de la géométrie des agents cartographiques) et de leurs relations avec les agents, ainsi que le principe de séparation entre déclaratif et procédural, prôné par IODA, doivent apporter un gain en généricité en permettant de décrire les contraintes et les actions qui peuvent les satisfaire indépendamment de leur résolution.

Nous avons donc entamé une collaboration sur ce sujet à travers un premier stage relativement exploratoire (Ala Atrash, 2011) qui visait à établir comment transposer logiciellement les relations de situation et d'encapsulation de PADAWAN dans l'architecture utilisée au COGIT. Ce stage a été suivi d'un deuxième, celui d'Adrien Maudet (2012), et poursuivi en thèse sous la direction de Cécile Duchêne, co-encadré par Guillaume Touya et moi-même.

L'objectif de cette thèse est de transposer le volet « comportemental » de PADAWAN dans le contexte de la généralisation cartographique. En particulier, l'objectif est de pouvoir représenter la structure d'une carte au moyen d'un graphe mixte (agents et environnements) et les comportements possibles sous forme de matrices d'interaction associées aux environnements. Cela nécessite d'assez grandes transformations.

— D'une part, la représentation des connaissances diffère notablement : les interactions à la IODA sont des règles conditions/actions, dont la téléonomie s'exprime dans les déclencheurs, tandis qu'un modèle à base de contraintes a nécessairement une structure téléologique (son objectif est de satisfaire les contraintes et

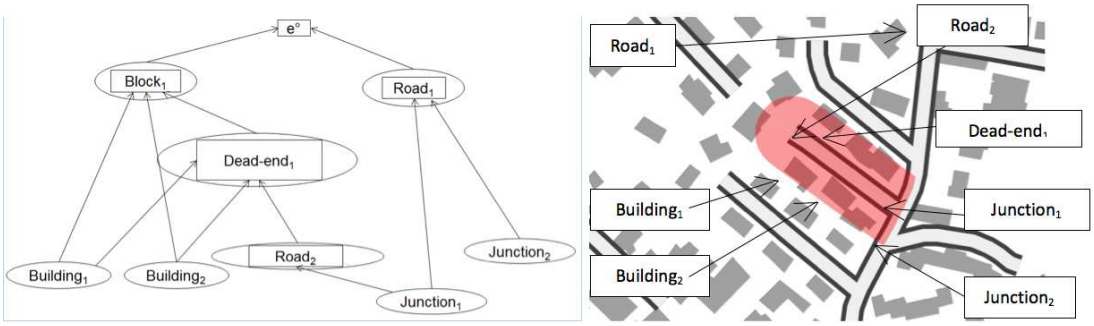


FIGURE 4.5 – Graphe mixte représentant les agents cartographiques et leurs environnements, détectés automatiquement pour une impasse (*dead-end*). L'impasse est placée dans un îlot (*block*) et héberge des bâtiments et une route. Cette route est elle-même en intersection avec la route principale qui borde l'îlot (emprunté à MAUDET *et al.* [7]).

il choisit des actions pour ce faire). Un travail de transposition de la notion d'interaction à la notion de contrainte a donc dû être mené.

— D'autre part, le moteur a dû être modifié substantiellement pour guider le choix des « interactions » à réaliser en fonction des avis émis par les contraintes et de leur degré d'insatisfaction.

À ce stage, ces modifications ont été implémentées dans la plateforme CartAGen du COGIT, et expérimentées avec succès dans le cadre d'un problème bien délimité, celui de la généralisation des impasses [7]. Une impasse, définie comme une zone d'habitation desservie par une route en cul-de-sac, peut en effet être vue comme un agent hébergé par un îlot, hébergeant des bâtiments et la route en cul-de-sac (fig. 4.5). De plus il faut tenir compte des routes voisines : ainsi, si plusieurs voies formant impasse se succèdent le long d'une route principale, l'espace situé entre les routes en cul-de-sac peut s'avérer trop petit pour représenter lisiblement tous les bâtiments de chacune des impasses, ce qui peut conduire à la suppression de l'une d'entre elles.

Bilan et perspectives

Les avantages de cette collaboration sont réciproques car la confrontation de nos méthodes de simulation aux problèmes fortement « situés » de la cartographie ont suggéré un certain nombre d'enrichissements pour PADAWAN. Ainsi, par exemple, le *pattern* d'**agent frontière** (stage d'Ala Atrash) correspond au cas d'un agent multi-situé, mais dont la suppression provoque systématiquement la fusion des environnements où il était situé. Ou encore, il se dessine également (thèse d'Adrien Maudet) un *pattern* **agent méso**, transposition dans PADAWAN du concept d'agent méso d'AGENT, qui correspond à un compartiment muni d'un certain nombre d'interactions destinées à gérer la résolution des contraintes des agents qu'il héberge.

Par ailleurs, nous avions dans IODA délibérément focalisé nos recherches sur un moteur de sélection d'interaction « réactif », c'est-à-dire guidé essentiellement par la structure téléonomique des interactions (déclencheurs et niveau de priorité). Nous avons, dans le contexte de la résolution de contraintes, l'opportunité d'étendre notre approche à une vision plus « cognitive », en mettant en œuvre des inférences ou de la planification par exemple. Nous évoquerons dans le chap. 6 quelques pistes sur le

rapprochement entre les domaines de la simulation et de la résolution distribuée de problèmes.

Enfin, l'application de PADAWAN au contexte cartographique permet également de progresser en direction de la réification de structures émergentes. En effet, si une partie de la structuration des agents et des environnements est donnée initialement par la décomposition « hiérarchique » de la carte (celle utilisée dans le modèle AGENT) avec une identification claire par exemple des îlots, certains groupes d'objets nécessitant un traitement global, donc formant un « niveau », sont détectés dynamiquement : c'est la cas par exemple des impasses composées de bâtiments autour d'une route ; les relations de situation et d'encapsulation sont donc créées en fonction des besoins identifiés par le système. Plus généralement, certains types d'agents méso peuvent faire l'objet d'une détection automatique [188, chap. C]. On peut donc envisager, à terme, une transposition de ces mécanismes en retour vers les techniques de simulation.

Chapitre 5

Informer les simulations par les données

> Sans généralisation, la prévision est impossible. Les circonstances où l'on a opéré ne se reproduiront jamais toutes à la fois. Le fait observé ne recommencera donc jamais ; la seule chose que l'on puisse affirmer, c'est que dans des circonstances analogues, un fait analogue se produira. Pour prévoir, il faut donc au moins invoquer l'analogie, c'est-à-dire généraliser.
>
> Poincaré [175, p. 169]

Dans ce que nous avons présenté jusqu'à présent, nous avons vu des thématiciens intervenir dans la construction des modèles pour y apporter l'expertise de leur domaine tant dans la définition des entités et des comportements que de leurs paramètres. Cette situation est largement représentative de la pratique de la simulation multi-agents. Toutefois, cette expertise humaine est comme toujours coûteuse à acquérir (au moins en temps), difficile à expliciter, parfois plus qualitative que quantitative, souvent entachée d'une grande part d'incertitude.

Or, les nouveaux domaines d'application des simulations multi-agents sont, nous l'avons dit, pourvoyeurs d'énormes volumes de données. Il devient donc peu à peu dangereux, pour la simulation, de prétendre construire des modèles sur la seule base d'une expertise humaine et de n'utiliser ces données qu'*a posteriori* pour valider les hypothèses initiales. Il semble plus judicieux d'élaborer des méthodes capables d'extraire des données les informations pertinentes pour la calibration et le paramétrage d'un modèle multi-agents relativement qualitatif, préalablement validé par les experts. Ces techniques doivent être adaptées à la fois à la nature de nos modèles et aux objectifs des simulations, de façon à reproduire les caractéristiques statistiques des entités modélisées et de leurs comportements, tout en conservant leur diversité, qui est le propre de l'approche multi-agents. On constate de fait que l'identification de

populations d'agents par des moyens statistiques devient une problématique saillante ces dernières années, que ce soit pour initialiser des simulations au moyen de paramètres appris, afin d'exprimer une diversité comportementale [137], ou pour détecter dynamiquement des groupes émergents [77].

Le problème auquel nous nous sommes attaqués dans ces travaux commencés tout récemment (courant 2012) consiste à doter des agents de caractéristiques individuelles différenciées construites à partir de traces d'activité enregistrées, de telle sorte que la population d'agents produise en simulation des traces similaires à celles des données. Nous avons pour cela fait un certain nombre d'hypothèses à la fois sur la forme de ces traces et sur les comportements qui en sont à l'origine, ce qui restreint le champ des techniques applicables mais permet d'aborder des domaines variés comme l'analyse de paniers d'achats, la génomique fonctionnelle ou l'étude des réseaux sociaux. Je vais d'abord expliquer la technique générale d'extraction de données que nous avons construite pour ce cadre, puis présenter l'exemple applicatif où elle a été testée, en l'occurrence la reconstruction de listes de courses à partir de tickets de caisse de clients dans un supermarché. De plus amples détails pourront être trouvés dans les articles correspondants [6, 24].

5.1 L'identification statistique de caractéristiques d'agents

5.1.1 Hypothèses et notations

La première de nos hypothèses est que certaines entités du système (réelles ou simulées) produisent des *traces d'activité* dont la forme est celle de **transactions** au sens de Agrawal et Srikant [55] : une transaction $\mathcal{T}$ est un *ensemble d'articles* (*items*), $\mathcal{T} = \{I_1, ..., I_n\}$. En règle générale les transactions sont de tailles différentes. Nous supposons également que de telles traces sont produites dans une certaine mesure par des *buts* qui peuvent être exprimés comme une *spécification partielle* des transactions souhaitées, ce que nous appelons un **prototype**. Les agents du système, réels ou simulés, tentent de réaliser leurs objectifs (représentés sous forme de transactions vaguement définies, imprécises ou incomplètes) dans le cadre d'un modèle comportemental donné et au sein d'un environnement particulier, ce qui peut entraîner des divergences entre les transactions effectivement réalisées par les agents et les prototypes visés.

La deuxième hypothèse est la possibilité d'*identifier des groupes d'agents* partageant un même prototype (c'est-à-dire les mêmes buts) au moyen d'une classification appropriée des traces d'activité ; ainsi, une analyse des transactions au sein de chacun de ces groupes doit permettre de reconstuire le prototype sous-jacent.

Notre dernière hypothèse est que chaque article d'une transaction est identifié au moyen d'un *tuple d'entiers strictement positifs*. Ces entiers sont destinés à représenter des propriétés décrivant les caractéristiques de l'article en question et dépendent du domaine d'application. Si κ caractéristiques sont jugées pertinentes, nous notons un article sous la forme $I = (f_1, ..., f_\kappa)$, avec $\forall i \in [1, \kappa], \quad f_i \in [1, \lambda_i]$. L'ensemble des articles représentables est donc $\mathcal{I}^\star = [1, \lambda_1] \times ... \times [1, \lambda_\kappa]$.

5.1.2 Des transactions aux prototypes

À partir d'une large base de données $\mathcal{D}$ de transactions enregistrées, le processus de recherche d'information consiste à construire un petit ensemble de prototypes visant à décrire chacun une « transaction cible idéale » et capable de caractériser des groupes de transactions similaires. Pour ce faire, nous avons également défini le concept d'**article prototype** : il s'agit également d'un tuple d'entiers, mais la valeur 0 est autorisée pour marquer *l'absence de spécification*, l'indifférence vis-à-vis de la caractéristique correspondante. De tels prototypes, construits à partir des données réelles, peuvent aussi être utilisés comme transactions cibles pour des agents simulés, puisqu'en règle générale seuls quelques éléments des articles souhaités sont spécifiés.

Un **prototype** $\mathcal{P}$ est simplement un ensemble d'articles prototypes, soit $\mathcal{P} = \{\widetilde{I_1}, ..., \widetilde{I_m}\}$, où les articles prototypes $(\widetilde{I_j})$ sont de la forme $(s_1, ..., s_\kappa)$ avec $\forall \ell \in [1, \kappa], s_\ell \in [0, \lambda_\ell]$. L'ensemble des prototypes représentables est donc $\mathcal{I} = [0, \lambda_1] \times ... \times [0, \lambda_\kappa]$.

5.1.3 Travaux connexes

Notre démarche s'apparente à la classification multi-label [211], dans la mesure où nous cherchons à identifier des groupes dont chacun est caractérisé par un prototype constitué de plusiers articles prototypes, lesquels pourraient être vus comme des étiquettes. Cependant, l'espace des étiquettes à considérer serait alors immense, ce qui rend de telles méthodes inapplicables.

D'autres méthodes viennent de l'analyse d'affinités, la plus répandue étant l'algorithme Apriori [55] destiné à identifier des règles d'association entre ensembles d'articles (sous la forme $X \to Y$ avec $X \cap Y = \emptyset$). À partir des co-occurrences d'articles dans les transactions, on peut ainsi construire des règles caractérisées par un *support* (proportion des transactions qui contiennent à la fois X et Y) et une *confiance* (probabilité conditionnelle de trouver dans une transaction les articles de Y lorsque ceux de X y sont déjà). Mais ces règles ne font que fournir une vue globale, statistique de l'ensemble des transactions, sans identifier de sous-groupes, ce qui n'est pas très adapté pour l'amorçage d'une simulation multi-agents.

Nous montrons dans la prochaine section comment construire effectivement les prototypes à partir des transactions.

5.2 Étapes du processus de recherche d'information

L'analyse des données enregistrées se décompose en deux étapes :

1. Les transactions de la base de données sont réparties en classes en fonction d'une mesure de distance entre transactions construite à cet effet, elle-même basée sur la distance entre articles.
2. Au sein de chaque classe, tous les articles présents sont à leur tour classés pour construire les articles prototypes pertinents, après quoi les prototypes possibles (unions d'articles prototypes) sont évalués au regard de toutes les transactions de la classe.

Notation	Description	Domaine
κ	nombre de caractéristiques des articles	$\mathbb{N}^*$
$\lambda_i \quad (i \in [1,\kappa])$	nombre de valeurs distinctes de la i^{e} propriété	$\mathbb{N} \setminus \{0,1\}$
f_i / $f_i(I)$	i^{th} caractéristique d'un article / de l'article I	$[1,\lambda] \subset \mathbb{N}$
$I = (f_1,...,f_\kappa)$	article	$\mathcal{I}^\star = [1,\lambda_1] \times ... \times [1,\lambda_\kappa]$
$\mathcal{T} = \{I_1,...,I_n\}$	transaction	$\mathbb{T} = \wp(\mathcal{I}^\star)$
$\mathcal{D} = \{\mathcal{T}_1,...,\mathcal{T}_N\}$	ensemble de transactions enregistrées	$\wp(\mathbb{T})$
$\widetilde{I} = (s_1,s_2,...,s_\kappa)$	article prototype	$\mathcal{I} = [0,\lambda_1] \times ... \times [0,\lambda_\kappa]$
$\mathcal{P} = \{\widetilde{I_1},...,\widetilde{I_m}\}$	prototype	$\wp(\mathcal{I})$
$\varsigma(f,f')$ / $\varsigma(s,s')$	similarité entre valeurs de caractéristiques	$\{0,1\}$
$\sigma(I,I')$ / $\sigma(\widetilde{I},\widetilde{I'})$	similarité entre articles / articles prototypes	$[0,1] \subset \mathbb{R}$
$J_{BM}(\mathcal{T},\mathcal{T}')$	similarité entre transactions selon « best-match Jaccard »	$[0,1] \subset \mathbb{R}$
$\Delta_{BM} = (\mathrm{d}_{i,j})$	matrice de distance entre toutes (N) les transactions	$[0,1]^{N^2}$
$\mathcal{C}_i = \{\mathcal{T}_1^i,...,\mathcal{T}_{n_i}^i\}$	i^e classe de transactions similaires	$\wp(\mathcal{D})$
$\phi_i(I)$	fréquence de l'article I dans les transactions de la classe $\mathcal{C}_i$	$[0,1] \subset \mathbb{R}$
ε,θ	seuils utilisé pour les articles rares et fréquents	$[0,1] \subset \mathbb{R}$
$\mathcal{R}_i,\mathcal{F}_i,\mathcal{O}_i$	articles rares, fréquents et ordinaires de la classe $\mathcal{C}_i$	$\wp(\mathcal{I}^\star)$
$(D^i_{j,k})$	matrice de distance entre articles ordinaires de la classe $\mathcal{C}_i$	$[0,1]^{\lvert\mathcal{O}_i\rvert^2}$
$\mathcal{P}_i^\star$	meilleur prototype pour la classe $\mathcal{C}_i$	$\wp(\mathcal{I})$

TABLE 5.1 – Résumé des notations utilisées pour l'analyse des transactions et la construction des prototypes.

Les notations introduites tout au long de la méthode sont résumées dans le tableau 5.1.

5.2.1 Mesure de similarité des articles

Dans le cas général, les articles qui apparaissent sur des transactions sont différents, même s'ils peuvent présenter des ressemblances. Étant donné que, par hypothèse, les articles prototypes sont indifférents à certaines caractéristiques, il n'est pas étonnant que des agents disposant du même prototype produisent des traces légèrement différentes. Aussi, la distance entre transactions ne peut s'appuyer seulement sur l'égalité des articles, sans quoi la classification risque d'être défectueuse. Nous avons donc choisi de moduler la comparaison entre transactions en prenant en compte une distance entre articles.

Une façon simple de procéder est de calculer une distance (ou inversement un indice de similarité) *à la Hamming* : si deux articles sont représentés par les tuples $I = (f_1,...,f_n)$ et $I' = (f'_1,...,f'_n)$, leur similarité est définie par :

$$\sigma(I,I') = \frac{1}{n}\sum_{i=1}^{n}\varsigma(f_i,f'_i), \quad \text{avec } \varsigma(f_i,f'_i) = 1 \text{ si } f_i = f'_i \text{ ou } f_i.f'_i = 0, \text{ et } \varsigma(f_i,f'_i) = 0 \text{ sinon.}$$

Cette définition s'applique aussi bien pour les articles concrets que pour les articles prototypes. De plus, elle peut être remplacée au besoin par toute mesure de similarité plus pertinente selon le domaine d'application.

5.2.2 Mesure de similarité des transactions

Pour comparer les transactions, nous nous sommes appuyés sur un indice utilisé couramment pour mesurer la similarité d'ensembles de tailles différentes : l'indice de

Jaccard [133]. Pour deux ensembles X et Y, il est défini comme suit :

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|}$$

Il existe un grand nombre d'indices de similarité et de mesures de distance [81, 71]. En pratique notre méthode est utilisable avec d'autres indices employés fréquemment, comme Ochiai [167], Kulczynski [135] et Sørensen-Dice [202, 91], qui peuvent être adaptés comme nous l'avons fait pour l'indice de Jaccard. Cette neutralité vis-à-vis des indices a été vérifiée empiriquement par la même procédure que celle que nous présentons dans la section 5.2.5.

Comme nous l'avons mentionné, l'indice de Jaccard ne peut être utilisé tel quel car il ne fait pas de différence entre des ensembles disjoints et des ensembles qui contiennent des éléments similaires mais néanmoins distincts. Nous avons donc construit une variante de cet indice, basée sur la distance entre articles. Elle consiste à calculer le score du *meilleur appariement possible* entre les articles des deux transactions à comparer. Cet indice, « *best-match Jaccard* » est noté J_{BM} et calculé au moyen de l'algorithme ci-dessous (algo. 1).

Algorithme 1 : Calcul de l'indice « best-match Jaccard »

Entrées : $\mathcal{T} = \{I_1, ..., I_p\}, \mathcal{T}' = \{I'_1, ..., I'_q\}$

1 calculer la matrice d'appariement $(\sigma_{i,j})$ i.e. $\forall i \in [1,p], j \in [1,q], \quad \sigma_{i,j} \leftarrow \sigma(I_i, I'_j)$

2 **pour** $k \leftarrow 1$ **à** $\min(p,q)$ **faire**

3 $\quad \mu_k \leftarrow \max(\sigma_{i,j})$

4 $\quad (i^\star, j^\star) \leftarrow$ un des couples $\underset{\{(i_0,j_0), \sigma_{i_0,j_0} = \mu_k\}}{\arg\min} \left(\sum_{i \neq i_0} \sigma_{i,j_0} + \sum_{j \neq j_0} \sigma_{i_0,j} \right)$

5 $\quad$ remplacer $(\sigma_{i,j})$ par la sous-matrice obtenue en supprimant la ligne $i^\star$ et la colonne $j^\star$

6 **fin**

7 $\mu_{BM} \leftarrow \sum_{k=1}^{\min(p,q)} \mu_k$

Résultat : $J_{BM}(\mathcal{T}, \mathcal{T}') \leftarrow \frac{\mu_{BM}}{p+q-\mu_{BM}}$ (μ_{BM} joue le même rôle que $|X \cap Y|$ dans l'indice de Jaccard classique)

Exemple. — Si l'on prend deux transactions $\mathcal{T}$ ={(1, 1, 2), (3, 5, 8), (13, 21, 34)} et $\mathcal{T}'$={(1, 1, 2), (3, 6, 8), (12, 13, 14), (1, 1, 34)}. Comme on a $\mathcal{T} \cap \mathcal{T}'$ ={(1, 1, 2)} seulement, on aurait $J(\mathcal{T}, \mathcal{T}') \approx 0.17$, tandis que l'on obtient $J_{BM}(\mathcal{T}, \mathcal{T}') = 0.4$, car plusieurs

$\sigma(I, I')$	(1, 1, 2)	(3, 6, 8)	(12, 13, 14)	(1, 1, 34)
(1, 1, 2)	**1**	0	0	0.6666667
(3, 5, 8)	0	**0.6666667**	0	0
(13, 21, 34)	0	0	0	**0.3333333**

Table 5.2 – Exemple de matrice d'appariement entre deux transactions, qui mesure la similarité entre tous leurs, utilisée pour calculer la valeur de J_{BM}. Les valeurs des μ_k sont en gras. On a ici $\mu_{BM} = 2$ d'où $J_{BM} = \frac{2}{3+4-2} = 0.4$.

articles de $\mathcal{T}$ et de $\mathcal{T}'$ sont assez proches comme on peut le voir sur la matrice d'appariement correspondante (tab. 5.2).

5.2.3 Classification des transactions

L'indice *best-match Jaccard* permet de calculer une **matrice de distance** $\Delta_{BM} = (\mathrm{d}_{i,j})$ entre *toutes les transactions* de la base : $\forall i \in [1,N], j\ i, \quad \mathrm{d}_{i,j} = 1 - J_{BM}(\mathcal{T}_i, \mathcal{T}_j)$. Une fois calculée, cette matrice peut être manipulée par des techniques de classification variées. Nous avons pour notre part appliqué un algorithme courant de classification hiérarchique (en l'occurrence celui implémenté en R par la bibliothèque `flashClust` [140]).

Le dendrogramme obtenu doit alors être divisé en K classes notées $\mathcal{C}_i = \{\mathcal{T}_1^i, ..., \mathcal{T}_{n_i}^i\} (1 \leq i \leq K)$. La valeur pertinente de K n'est pas connue *a priori* : dans nos premières expériences, nous avons estimé empiriquement la hauteur de coupe la plus judicieuse à partir de jeux de données dont le nombre de prototypes était connu [6, 24] ; depuis, grâce au stage de Guillaume Dauster (2013) nous avons amélioré cette technique en utilisant une évaluation intrinsèque de la qualité de la classification, le critère de **silhouette** défini par Rousseeuw [185].

5.2.4 Reconstruction du prototype

Construire un prototype pour la classe $\mathcal{C}_i$ consiste à trouver un ensemble $\mathcal{P}_i$ de tuples d'entiers (éventuellement nuls) qui obtient le meilleur score d'appariement (*via* l'indice *best-match Jaccard*) avec les n_i transactions de la classe. Il est clair qu'un ensemble composé de tuples nuls $(0,0,...,0)$ peut s'apparier à toutes les transactions, mais aussi à celles des autres classes, et de plus conduirait les agents des buts complètement aléatoires au lieu de ceux observés dans la classe. Le prototype doit donc être aussi spécifié que possible en ne généralisant (en n'utilisant de 0) que sur les caractéristiques réellement neutres. Pour cela, nous procédons d'abord à une *analyse de fréquence* des articles, en calculant :

$$\forall I \in \bigcup_{\mathcal{T} \in \mathcal{C}_i} \mathcal{T}, \quad \phi_i(I) = \frac{1}{n_i}.|\{\mathcal{T} \in \mathcal{C}_i, I \in \mathcal{T}\}|$$

Nous utilisons deux valeurs de seuil ε and θ pour répartir les articles qui apparaissent dans les transactions de la classe $\mathcal{C}_i$ en trois sous-groupes : $\mathcal{R}_i = \{I | \phi_i(I) \leq \varepsilon\}$ (les articles rares), $\mathcal{F}_i = \{I | \phi_i(I) > \theta\}$ (les articles fréquents) et $\mathcal{O}_i = \{I | \varepsilon < \phi_i(I) \ \leq \ \theta\}$ (les articles ordinaires).

Nous faisons aussi l'hypothèse que les articles rares sont présents dans la transaction « par hasard », et par conséquent ils sont écartés du prototype à construire. Le choix d'une valeur appropriée pour ε dépend évidemment du domaine d'application ; dans les expériences de validation décrites plus loin (§5.2.5), nois avons utilisé $\varepsilon = \frac{1}{n_i}$, autrement dit les articles éliminés sont ceux qui n'apparaissent que dans une seule des transactions de la classe. Inversement, les articles fréquents peuvent être considérés comme un but commun à tous les agents de la classe, donc sont conservés tels quels dans le prototype (même remarque pour θ).

Restent les articles ordinaires, qu'il faut à nouveau classifier afin de pouvoir déterminer, pour des articles similaires, quelles sont les caractéristiques qui doivent avoir

une valeur particulière et quelles sont celles qui sont indifférentes. Nous calculons donc une matrice de distance entre articles :

$$(D^i_{j,k}) \quad \text{avec} \quad \forall I_j, I_k \in \mathcal{O}_i, D^i_{j,k} = 1 - \sigma(I_j, I_k)$$

que nous utilisons pour construire un dendrogramme des articles. À nouveau, le nombre de classes d'articles similaires, K_i, n'est pas connu *a priori*. Nous appliquons donc l'algorithme ci-dessous (algo. 2) pour déterminer le meilleur prototype pour la classe $\mathcal{C}_i$.

Algorithme 2 : Construction d'un prototype

pour $K_i \leftarrow 1$ **à** $|\mathcal{O}_i|$ **faire**
  calculer les K_i classes d'articles similaires : $\chi_j = \{I^j_1, ..., I^j_{m_j}\} \quad (1 \leq j \leq K_i)$
  pour $j \leftarrow 1$ **à** K_i **faire**
    calculer un article prototype pour la classe χ_j : $\widetilde{I_j} = (s_1, ..., s_\kappa)$ avec $\forall \ell \in [1, \kappa], \quad s_\ell \leftarrow f_\ell . \prod_{I, I' \in \chi_j} \varsigma(f_\ell, f'_\ell)$ ($s_\ell \leftarrow 0$ ssi la ℓ^e caractéristique diffère pour des articles de la classe, sinon on prend la valeur de cette caractéristique)
  fin
  construire le prototype candidat pour $\mathcal{C}_i$ et K_i : $\mathcal{P}_{i,K_i} \leftarrow \mathcal{F}_i \cup \left(\bigcup_{j \in [1,K_i]} \widetilde{I_j}\right)$
  calculer le score de ce prototype sur toutes les transactions de la classe : $v(\mathcal{P}_{i,K_i}) \leftarrow \frac{1}{n_i} . \sum_{k=1}^{n_i} J_{BM}(\mathcal{P}_{i,K_i}, \mathcal{T}^i_k)$
fin
Résultat : $\mathcal{P}^\star_i = \underset{\mathcal{P}_{i,Ki} \ \forall K_i \in [1,|\mathcal{O}_i|]}{\arg\max} v(\mathcal{P}_{i,K_i})$

5.2.5 Validation du processus

Comme nous l'avons expliqué, les prototypes ont vocation à être utilisé durant la simulation pour produire de nouvelles transactions reflétant l'activité des agents. Dans la mesure où ces agents exécutent leurs comportements d'une façon autonome sur la base de ces prototypes, mais dans un environnement dynamique, les transactions obtenues en simulation n'ont pas de raison de coïncider parfaitement avec celles des données d'origine. Pourtant, nous devons nous assurer que la simulation est capable de reproduire *le même genre de traces* que celles observées dans la réalité. Un bon critère consiste à analyser les traces simulées par les *mêmes techniques* que les traces réelles, et comparer les prototypes obtenus : à ce niveau d'abstraction, il doit y avoir identité.

Toutefois avant même de mener ce genre d'étude sur les résultats d'une simulation multi-agents, il fallait s'assurer de la robustesse du processus d'analyse lui-même. Cela a été fait au moyen de nombreuses simulations stochastiques, consistant à partir d'un ensemble de prototypes aléatoires, à « instancier » ces prototypes en transactions (en remplaçant les o par des entiers strictement positifs, et en ajoutant du bruit), puis à reconstruire les prototypes. Deux critères d'évaluation sont utilisés :

— une *matrice de confusion*, qui donne la proportion des transactions issues d'un prototype $\mathcal{P}_i$ qui ont été regroupées dans la classe $\mathcal{C}_j$: à une permutation des indices près, on doit obtenir une correspondance exacte (toutes les transactions issues de $\mathcal{P}_i$, et elles seules, doivent constituer la classe $\mathcal{C}_i$) ;
— une *matrice de distance* entre les profils initiaux $\mathcal{P}_i$ et les profils reconstruits $\mathcal{P}'_j$, distance calculé grâce à l'indice *best-match Jaccard* (J_{BM}) : là encore, à une permutation des indices près, on doit avoir $J_{\text{BM}}(\mathcal{P}_i, \mathcal{P}'_i) = 1$.

Les détails de cette évaluation figurent dans [6, 24] et permettent d'affirmer que le processus est résistant à plusieurs formes de perturbations qui sont susceptibles d'apparaître dans les traces d'activité réelles :
— l'ajout, dans chaque transaction, d'articles ne figurant pas dans le prototype d'origine (traces d'actions effectuées sans avoir été dirigées par des buts) ;
— la suppression aléatoire, dans chaque transaction, d'une petite proportion d'articles qui auraient dû y figurer (buts non satisfaits) ;
— la présence de transactions aléatoires, c'est-à-dire qui ne sont déterminées par aucun prototype (actions désordonnées d'agents qui n'agiraient pas conformément à des buts homogènes aux autres).

Je vais maintenant présenter le contexte applicatif dans lequel ces méthodes ont été mises en œuvre.

5.3 La simulation de clients statistiquement réalistes

Pour évaluer notre méthode, nous nous sommes placés dans le cadre de simulations de comportements de clients dans un magasin, sur la base de nos expériences antérieures dans ce domaine [10, 38]. L'approche multi-agents est déjà utilisée dans le domaine de la distribution [194, 197], dans la mesure où la modélisation fine des comportements individuels permet d'étudier l'impact d'actions commerciales ou de changements dans l'aménagement de la surface de vente. L'utilisation de données pour accroître le réalisme des clients simulés consitue le projet INBASU (pour « *INdividual-BAsed simulation of SUpermakets* ») qui s'intéresse à l'élaboration d'outils d'aide à la décision en marketing sur la base de simulations multi-agents.

Environnement et comportements

Dans de précédents travaux [10], nous avons élaboré une simulation de supermarchés où les agents sont situés dans un environnement spatialement réaliste. Ce caractère situé est nécessaire pour transformer une simulation *ad hoc* en un véritable outil d'aide à la décision, capable de prédire comment les clients réagissent aux changements de l'organisation spatiale du magasin, aux événements commerciaux ou encore à la pression concurrentielle. Néanmoins, il faut aussi pour cela garantir la fidélité des profils de consommateurs, afin d'assurer le réalisme non pas uniquement des actions effectuées par les agents simulés, mais également de la diversité de leurs objectifs d'achats.

Le modèle de comportement sur lequel nous nous appuyons est voisin de celui de FORMAT-STORE (§3.2.1) avec quelques simplifications (tab. 5.3). Le comportement

Sources \ Cibles	∅	Client	Article	Caisse	FileAttente	Porte
Client	Errer (0) ChoisirDest (1)		AllerVers (2 ; 10) Prendre (4 ; 1)	AllerVers (3 ; 10)	Entrer (5 ; 2) AllerA (6 ; 1) Sortir (7 ; 1)	Quitter (8 ; 1)
Article		Informer (1 ; 10)				
Panneau		Informer (1 ; 10)				
Caisse	Ouvrir (10) Fermer (10)	Informer (1 ; 15) Encaisser (7 ; 1)			Traiter (8 ; 1) Interrompre (9 ; 1)	
Porte	CréerClient (1)	Informer (1 ; 10)				

TABLE 5.3 – Matrice d'interaction qui définit le comportement des agents pour la simulation de clients dans un supermarché (cf. [24]).

générique des clients simulés, été validé par des experts en marketing, est ici *considéré comme figé*. Les clients ont un même comportement d'ensemble, mais *diffèrent dans leurs besoins*, aussi sont-ils dotés d'une *liste de courses* qui spécifie, de façon plus ou moins détaillée, quels articles ils sont susceptibles d'acheter dans le magasin. Les transactions enregistrées sont alors des *tickets de caisse*, contenant des *articles*, tandis que les prototypes sont les *listes de courses*.

Expérimentations multi-agents

Dans un premier temps, afin de comparer les simulations multi-agents aux simulations stochastiques utilisées pour évaluer la robustesse du processus d'analyse des données, nous avons utilisé les mêmes prototypes, avec des agents connaissant la position des rayons dans le magasin. Dans ces conditions, les transactions simulées sont capables de reproduire les classes et les prototypes de départ.

Ces résultats changent légèrement lorsque l'on modifie les conditions environnementales ou certains paramètres de simulation. Ainsi, donner aux agents une *limite temporelle* pour effectuer leurs achats, lorsque ceux-ci ne connaissent pas l'agencement spatial du magasin, peut avoir pour effet que certains agents sortent du magasin sans avoir trouvé tous les articles de leur liste de courses. De la même façon, on peut faire en sorte que certains articles ne soient pas présents en rayon ou mal signalés. Introduites avec modération, ces modifications n'ont pas d'effet sur *l'identification des classes de transactions* (grâce à la robustesse du processus vis-à-vis des articles manquants). En revanche, elles peuvent changer les prototypes reconstruits à partir des transactions simulées : dans une simulation stochastique, faire « disparaître » un article des transactions n'a statistiquement pas d'impact car chaque article a une probabilité uniforme d'être manquant. À l'inverse, lorsque les agents sont soumis au caractère éminemment spatial de leur environnement et doivent y trouver les informations correctes pour atteindre leurs buts, il peut apparaître un biais statistique en défaveur des articles les moins faciles à trouver. L'observation des chemins suivis par les clients dans le magasin (cf. fig. 5.1) met en effet en évidence l'existence de « points chauds » très fréquentés, où les articles sont donc trouvés facilement, ainsi que de « points froids » où la fréquentation est faible : les produits manquants sont donc souvent les mêmes, de sorte que les prototypes reconstruits dans ces conditions constituent un sous-ensemble des prototypes d'origine.

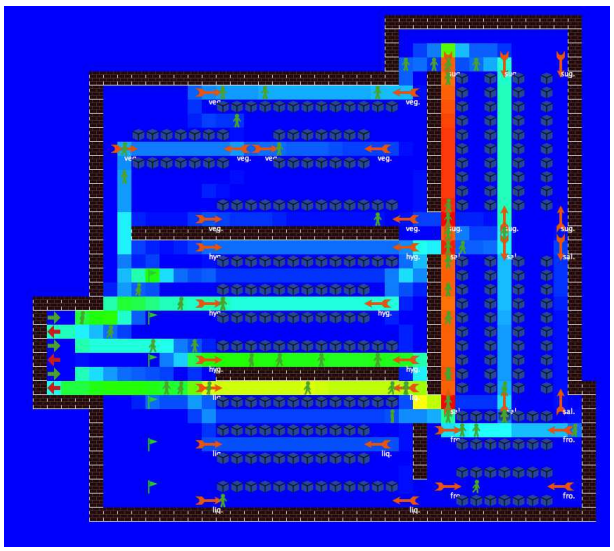


FIGURE 5.1 – Tracé des chemins suivis par les clients simulés dans le magasin virtuel. Les couleurs représentent la fréquentation relative de chaque zone de l'environnement (du bleu, minimal au rouge, maximal).

Ce phénomène met en lumière le côté crucial de la mise en situation spatiale des simulations, et illustre bien comment utiliser ces outils pour l'aide à la prise de décision quant au placement des articles en rayon, l'agencement du magasin, la signalisation, les événements commerciaux, etc.

Les expériences en cours (notamment le stage de Guillaume Dauster, 2013) visent à étudier de façon systématique l'impact de ces informations spatiales sur les changements qui surviennent dans les prototypes. Par ailleurs nous travaillons à l'intégration de bases de données de tickets réels et d'environnements de plus grande taille reproduisant les caractéristiques du magasin où ces tickets ont été collectés. Une grosse part des efforts porte sur la récupération des informations de positionnement des articles et des dispositifs d'information.

5.4 Bilan

La conception de simulations puisant leur information au plus près des données réelles, en réduisant l'expertise humaine lorsque c'est possible, est évidemment un objectif de longue haleine, tout particulièrement si l'on vise l'élaboration de méthodes indépendantes du domaine d'application. Nous donnerons dans le chapitre suivant quelques pistes dans ce sens.

D'ores et déjà, comme nous l'avons montré, les algorithmes d'exploration de données que nous avons définis, au lieu de servir à définir un « comportement moyen » valable pour l'ensemble de la population, permettent 1° d'identifier diverses populations d'agents sur la base des traces laissées par leurs comportements, 2° d'associer automatiquement à ces groupes une représentation abstraite de leurs buts (prototypes), et 3° d'engendrer en simulation des populations d'agents différenciées mais présentant statistiquement les mêmes caractéristiques que les populations d'origine.

La similarité entre résultats de simulation et données sources peut être mesurée finement, en utilisant les mêmes moyens que pour le processus de construction de connaissances.

Notre méthode a montré son efficacité dans un cadre applicatif où nous bénéficions déjà d'une assez longue expérience. On notera que, contrairement à ce que visent nombre d'études en marketing, nous ne cherchons nullement à identifier des segments de clientèle « classiques », mais simplement à reproduire en simulation une « photographie » d'une population : l'instant du cliché, comme la durée de pose, c'est-à-dire le moment et la durée d'enregistrement des transactions, conditionnent évidemment les caractéristiques qui seront ensuite données aux agents. Ainsi, la même méthode peut, à volonté, intégrer des variations saisonnières, géographiques, etc. selon les données recueillies.

Dans le domaine du marketing par exemple, ces recherches permettent, en renforçant le réalisme des agents simulés, d'envisager la réalisation d'outils d'aide à la décision de plus en plus fiables, d'une part en étudiant comment des changements hypothétiques dans la gestion de l'environnement ou dans les événements commerciaux sont susceptibles d'affecter une clientèle connue, et d'autre part, en anticipant les réactions possibles de nouveaux profils de consommateurs à un environnement donné. Dans des domaines scientifiques, elles fournissent une automatisation accrue de la conception et de la calibration des modèles, autorisant ainsi les thématiciens à se focaliser sur les aspects les plus qualitatifs (définition et affectation des interactions notamment).

Enfin, il nous reste à mettre nos méthodes à l'épreuve d'autres disciplines, par exemple celles qui utilisent déjà des méthodes de classification proches (classification multi-label) ou des techniques d'identification de paramètres, comme la génomique fonctionnelle [61] ou l'écologie [62].

Chapitre 6

Projet de recherche

> Le problème important — en matière de philosophie du langage — n'est pas celui de la vérité [...] mais bien celui de l'acceptabilité sémantique, qui définit le monde des « possibles ». [...] Ici, on ne cherchera pas à convaince, mais à susciter des représentations, et à étendre l'intelligibilité du monde.
>
> Thom [206]

Les thèmes que j'ai évoqués tout au long de ce mémoire nous ramènent toujours au même constat : l'étude des systèmes complexes à large échelle au moyen de simulations multi-agents nécessite l'élaboration de méthodes et de techniques originales, capables tout à la fois de *simplifier le processus de conception* autant que possible du côté du thématicien, que de traiter algorithmiquement des situations de plus en plus complexes avec *efficacité et fiabilité*.

L'approche orientée interactions, conçue dans ce but, est encore largement perfectible, et les avantages qu'elle procure en termes d'intelligibilité et de révisabilité des modèles, méritent qu'on travaille à son évolution. Mais celle-ci ira forcément de pair avec les voies qui se sont ouvertes ces dernières années, en particulier dans le cadre de la simulation multi-niveaux.

6.1 Développements de l'approche orientée interactions

L'approche orientée interactions, sous sa forme actuelle, laisse de côté un certain nombre de problèmes, soit parce que nous avons de bonnes raisons de penser qu'ils peuvent être résolus par une modélisation adéquate, soit parce qu'ils étaient provisoirement écartés par les hypothèses que nous avions choisies.

Les modèles temporels

La première de ces hypothèses est celle du *temps discret*, et plus encore le choix qu'une interaction se déroule tout entière au cours d'un pas de temps. Cette exigence peut sembler forte mais elle est souvent à la base de nombreux systèmes réactifs, et Kubera [134] a montré que l'on pouvait décomposer des actions longues en une succession d'interactions atomiques. Il est possible également de doter les agents du degré suffisant de mémoire pour sélectionner comme cible un agent qui était déjà leur cible au pas de temps précédent, par exemple afin d'assurer la persistance des comportements [213].

En revanche, il est plus difficile dans le cadre IODA actuel d'exprimer le fait que les *actions*, au sein d'une interaction, occupent un certain laps de temps. Pour étendre les possiblités de IODA, il faudrait commencer par doter les primitives d'une durée, après quoi il devrait être possible d'adapter des procédés existants. On peut envisager par exemple de décrire la séquence d'actions à la façon d'une *tâche* au sens de Drogoul [95], mais le *modèle à temporalité* proposé par Payet *et al.* [172] nous semble plus prometteur, dans la mesure où il conjugue des mécanismes empruntés à la fois à l'approche à temps discret et à l'approche à événements discrets, et se prête sans doute plus facilement à la gestion du temps pour des systèmes multi-niveaux.

Des moteurs alternatifs

Nous avons également fait le choix dans IODA de construire des simulations à partir d'un moteur « réactif », au sens où une interaction peut avoir lieu dès que ses condition et déclencheur sont vérifiés par les agents candidats, moyennant de respecter la garde de distance. C'est aux priorités qu'il appartient ensuite de définir une relation d'ordre entre les interactions réalisables par un agent source.

Comme nous l'avons mentionné à propos des travaux menés en collaboration avec le COGIT (§4.4), d'autres sortes de moteurs peuvent être construits. En particulier, il est possible d'utiliser le déclencheur d'une interaction pour décrire explicitement son but, ce qui autorise ensuite un moteur à chaînage pour décider des interactions à effectuer en vue d'un but global.

Syntaxe et interactions

Si l'approche orientée interactions procure une grande intelligibilité aux modèles de comportements, c'est en somme parce qu'une interaction peut être vue comme un *verbe*. La source de l'interaction en est le *sujet*, tandis que la (les) cible(s) peu(ven)t être vue(s) comme un *complément d'objet* ou comme un *complément d'agent*, selon que le verbe est exprimé à la voix active (Manger) ou à la voix passive (ÊtrePorté). Les interactions réflexives peuvent être interprétées de la même façon, soit comme un verbe réflexif (S'Orienter), soit comme un verbe intransitif (Mourir).

Cet arrière-plan syntaxique donne des pistes intéressantes pour étendre IODA, par exemple en introduisant des rôles supplémentaires à ceux de source et de cible, comme par exemple *outil* (complément circonstanciel : le Bûcheron effectue Couper sur l'Arbre *au moyen d'une* Hache). Le risque sous-jacent est d'augmenter sensiblement la

complexité algorithmique de la recherche des interactions réalisables [28] : il faut donc d'abord évaluer le gain d'expressivité que l'on attend de tels ajouts.

Sémantique et interactions

En tant que méthode de conception de simulations, IODA a été conçue pour minimiser le contact des thématiciens avec le code, tout particulièrement avec les parties les plus « sensibles » de ce code comme l'ordonnanceur de la simulation. En utilisant JEDI-Builder ou IODA-NetLogo, il reste peu de choses à écrire : les primitives de chaque famille d'agents et de l'environnement. C'est peut-être encore trop car nombre de thématiciens sont peu ou pas formés à l'algorithmique.

Une piste serait de spécifier la sémantique des primitives d'une façon plus proche du langage naturel, pour ensuite les traduire de façon plus ou moins automatique dans un langage de programmation. Mais il me semble que les solutions les plus réalistes sont plutôt de nature graphique et constructionniste. Ainsi, par exemple, Bersini [65] défend l'usage d'UML comme terrain commun pour définir non seulement les entités d'un modèle de simulation mais également leur comportement (notamment au moyen des diagrammes de séquences et d'états). On a lieu de craindre que la conception d'un modèle UML complet soit aussi difficile pour un thématicien que celle d'un modèle computationnel ; néanmoins la perspective change notablement s'il ne s'agit de définir que les primitives, dont la complexité individuelle est sans commune mesure avec celle de l'ensemble. Parmi d'autres alternatives intéressantes, le langage Scratch [147], conçu en vue d'enseigner la programmation et l'algorithmique à de jeunes enfants, fournit une syntaxe visuelle pour représenter des branchements, des boucles, des événements, des fonctions, etc. de sorte que l'on peut construire des programmes par assemblage et composition d'éléments visuels.

Dans un cadre plus spécifiquement multi-agents, la « programmation situationnelle » proposée par Michel *et al.* [156] semble encore plus prometteuse : elle vise à se détacher de l'algorithmique en demandant au concepteur d'élaborer des *situations* à partir de percepts de bas niveau, et de paramétrer un plan, le tout sous une forme également graphique. Les limitations inhérentes à cette approche (notamment une assez forte dépendance au domaine) pourraient certainement être levées dans le cadre de la forte séparation déclaratif/procédural imposée par IODA.

6.2 Simulation multi-niveaux et utilisation de données

Au vu des recherches menées dans ce domaine depuis quelque temps, il me semble qu'on peut esquisser trois sujets majeurs qui se développent en parallèle, et dont l'interdépendance promet des avancées importantes pour la simulation multi-agents dans les années à venir :

1. D'abord, on peut signaler avec Morvan [163] l'explosion de l'intérêt (sinon des recherches) pour la simulation multi-niveaux. Cela tient à la nature même des phénomènes que la simulation multi-agents cherche désormais à étudier, mais il est certain que le « découpage » d'une simulation monolithique en sous-systèmes est en outre une bonne façon de réduire la complexité du phénomène

à modéliser, d'accroître la modularité des simulations et par là même la révisabilité des modèles — pour peu qu'on dispose de méthodes génériques pour ce faire.

2. Ensuite, il devient crucial, pour la calibration et le paramétrage des simulations, de disposer de techniques de recherche automatique d'informations dans les immenses volumes de données issus de ces phénomènes. L'enjeu ne se réduit pas à l'extraction de statistiques simples, mais bien à l'identification et à la construction de connaissances pertinentes dans un cadre multi-agents.
3. Enfin, on voit également se développer une étude détaillée des possibilités de distribution physique du calcul sur des architectures physiquement distribuées ou simplement parallèles [155, 80, 151]. Cela répond à la montée en échelle des calculs à effectuer, mais ouvre également des perspectives de modélisation de type « champ » [155] qui n'étaient guère abordables jusqu'à présent, et par suite la possibilité de jeter des ponts vers des approches utilisant le continu.

Ce n'est sans doute pas un hasard si ces trois sujets émergent de façon concomitante : ils naissent des mêmes besoins. Mais en outre, ils sont susceptibles de s'assister mutuellement :

— la simulation multi-niveaux utilise des techniques de comparaison et de classification des agents pour détecter les structures émergentes [209, 120, 160, 78, 77], de sorte que les travaux sur les informations qui peuvent être extraites des données pour la calibration des comportements ne peuvent que faire progresser les recherches sur l'identification et l'agentification de macro-agents ;
— réciproquement, il est sans doute plus facile de traiter et de rendre intelligibles de grosses masses de données si l'on est capable de les projeter sur différents niveaux d'analyse ;
— au fur et à mesure que les simulations gagnent en complexité, elles sont elles-mêmes susceptibles de produire des volumes de données considérables, qu'il faut également être capable de représenter d'une façon synthétique et en rapport avec la forme multi-agents des modèles ;
— la simulation multi-niveaux nécessite souvent une montée à l'échelle, de sorte que l'utilisation de méthodes de calcul parallèle permet, comme le montre Michel [155], de traiter des problèmes de grande ampleur en un temps raisonnable — de fait on commence à voir des plateformes de simulation « classique » prendre le tournant du calcul haute performance (*High Performance Computing*) [82].

Le second volet de mon programme de recherche visera donc à prolonger les travaux amorcés sur la simulation multi-niveaux, en y intégrant les problématiques d'extraction de connaissances à partir des données.

Problématiques liées au multi-niveaux

Pour ce qui est de la simulation multi-niveaux, nous disposons avec PADAWAN d'un cadre de modélisation tout à fait adéquat. Il reste maintenant à travailler sur les points suivants :

— se servir de régularités observées dans des groupes d'agents pour les agréger automatiquement en un agent compartiment, doter celui-ci d'un comportement macroscopique (sous la forme d'interactions) qui reflète le comportement collectif des agents qu'il héberge, qui eux-mêmes peuvent ensuite être éliminés si c'est nécessaire (par exemple pour réduire les coûts computationnels) — on notera que ces préoccupations rejoignent les travaux autour de la réification de structures émergents mais supposent également un travail en amont sur l'expression d'une « sémantique » des interactions ;
— inversement, sur la base par exemple des techniques employées pour générer des populations diversifiées de clients (§5.3) ou pour anticiper l'évolution de structures agrégées [173], élaborer des méthodes qui permettent, à partir du comportement d'un agent, de recréer automatiquement le système sous-jacent, en générant des micro-agents qui reflètent un état possible de ce système et vont ensuite interagir pour produire collectivement un comportement équivalent.

Problématiques liées au traitement des connaissances

Jusqu'à présent nous avons posé une hypothèse forte sur la forme des données. En particulier, une *transaction* ne garde pas de trace de l'ordre dans lequel l'agent a satisfait (ou cherché à satisfaire) ses buts. Or, il arrive fréquemment dans les systèmes naturels qu'on dispose d'informations *chronologiques* : une séquence d'événements accomplie par les agents. Les traces sont alors ordonnées, voire caractérisées par une durée. Dans les systèmes artificiels, ces informations peuvent évidemment être recueillies à des fins d'analyse.

Afin de généraliser l'approche que nous avons proposée, une prochaine étape consistera à mettre au point des méthodes capables de traiter des traces chronologiques. Un certain nombre de pistes sont envisageables pour cela, la plus prometteuse étant semble-t-il à chercher du côté de *l'analyse de processus* (*process analysis*) ou de workflows [53, 52].

Synthèse

On voit ainsi se dessiner une double approche pour analyser des comportements collectifs et leurs rapports aux comportements individuels :

— une approche **synchronique** qui, étant donné une certaine fenêtre temporelle, détermine *les propriétés et les buts* de groupes d'agents similaires, de façon à pouvoir agréger un groupe en un macro-agent ou inversement, dissoudre un agent en un système d'entités plus fines ;
— une approche **diachronique** qui, pour un groupe d'agents donné, s'intéresse à la *succession de ses actions* pour identifier un comportement type, voire inférer les interactions sous-jacentes.

Sans vouloir à toute force laisser penser qu'il n'y point de salut en dehors de l'approche orientée interactions, il me semble néanmoins qu'un tel programme ne peut être mis en œuvre sans s'appuyer sur des principes assez similaires, en particulier une séparation déclaratif/procédural et une séparation entre entités, comportements

et espaces (ce que nous traduisons dans IODA/PADAWAN par agents, interactions et environnements).

Pour rester dans la « ligne » IODA, nous pourrions appeler ce projet « KENOBI » (pour : *Knowledge Extraction from Natural Observations to Build Interactions*).

6.3 Par-delà la simulation

De la simulation à la résolution de problèmes

Il me faut également mentionner une autre piste de recherche qui s'est ouverte avec la thèse d'Adrien Maudet sur la généralisation cartographique (IGN). Les transformations introduites dans le cadre de PADAWAN pour passer de la simulation multi-niveaux à la résolution de contraintes dans un cadre également multi-niveaux, offrent en effet une opportunité de concevoir une méthode générale de résolution de problèmes au moyen d'un SMA multi-niveaux.

Une telle méthode a un intérêt lorsque le problème est par nature multi-niveaux, comme c'est le cas pour résoudre des contraintes cartographiques, mais aussi lorsque les entités impliquées dans le problème participent à des relations de composition (par exemple pour la gestion de problématiques énergétiques à l'échelle d'un quartier ou d'une ville). Mais elle pourrait également s'appliquer s'il est possible de décomposer le problème en sous-problèmes partiellement disjoints, n'impliquant que des ensembles plus réduits d'entités : autrement dit, si une forme d'organisation peut-être déduite des relations entre les entités du problème.

Cette perspective devrait être étudiée par Adrien au cours de sa dernière année de thèse. Elle permettrait de rapprocher à nouveau le domaine de la simulation multi-agents de celui de la résolution distribuée de problèmes (pouvant être eux-mêmes distribués ou non). Ce projet qui faisait partie du bagage originel des SMA, par exemple à travers l'Éco-résolution [96], a depuis lors fait l'objet de très peu de travaux (par exemple [76]) en-dehors de contextes applicatifs.

De la simulation au contrôle de SMA

Par ailleurs, utiliser des interactions pour décrire les comportements indépendamment des agents présente des avantages qu'il serait intéressant de transposer vers le domaine de la *conception d'applications* multi-agents, ou encore vers le *contrôle* de SMA (agents logiciels ou robotiques). Il faudrait sans doute pour cela remplacer l'algorithme de sélection d'interaction, centralisé au niveau de l'ordonnanceur du simulateur, par des protocoles entre groupes d'agents susceptibles d'interagir de façon asynchrone.

Nous comptons entamer dans les prochains mois une étude exploratoire de ces questions, dans un cadre de robotique mobile collective. Il s'agira principalement de valider expérimentalement la possibilité de transposer l'approche orientée interactions dans ce domaine et d'étudier les transformations techniques mais aussi méthodologiques à y apporter.

6.4 ... et encore au-delà ?

> C'est un instrument mathématique qui crée la science physique contemporaine comme le microscope crée la microbiologie. Pas de connaissances nouvelles sans la maîtrise de cet instrument mathématique nouveau.
>
> Bachelard [59, p. 58]

Au début du siècle précédent, comme le remarque Bachelard [59], les mathématiques ont joué un rôle clef en physique, non pas tant par l'efficacité calculatoire de méthodes analytiques nouvelles, mais plus fondamentalement parce que les *concepts* mathématiques de l'époque ont *forgé* de toutes pièces ceux de la physique, relativiste ou quantique. Non seulement les modèles de la physique s'expriment au moyen des outils mathématiques les plus avancés, mais ils n'ont même plus de rapport (autre que linguistique) avec les réalités perceptibles du quotidien : ainsi par exemple les « particules » n'ont plus de matérialité et se réduisent à une fonction d'onde.

Si Bachelard ne cache pas son admiration pour cette hégémonie créatrice des mathématiques vis-à-vis d'autres disciplines, à l'inverse un mathématicien contemporain comme Thom [206] regrette la *perte d'intelligibilité* de la nature qui s'ensuit : selon lui, la physique quantique a abandonné le projet de rendre le monde compréhensible. Son propre projet consistait, en réaction, à tenter de rendre opérationnels des concepts plus qualitatifs, issus de la physique d'Aristote. Force est de constater que les méthodes qu'il a employées ne sont, hélas ! guère intelligibles qu'à des mathématiciens de haut vol...

Peut-être est-il possible de réconcilier ces deux points de vue. L'informatique en général, la simulation multi-agents en particulier, sont des « sciences de l'artificiel » [198] qui, comme les mathématiques, *créent leurs propres objets d'étude*. Mais, contrairement aux mathématiques, l'informatique peut également *restaurer de l'intelligibilité* en donnant une réalité structurelle et algorithmique à des concepts que la démarche scientifique, depuis Galilée, a disqualifiés mais que l'on peut souhaiter préserver pour leur capacité à rendre la nature intelligible. Ainsi, par exemple, si l'on ne croit plus guère à la réalité des Universaux, la programmation par objets consiste à reconstruire un monde dans lequel ils ont une valeur opérationnelle, parce qu'ils font sens. Ainsi, face aux sciences des systèmes complexes, il me semble que nos efforts doivent tendre à faire jouer à la simulation informatique un rôle structurant et créateur de concepts.

Troisième partie

Pièces annexes

Mon entreprise n'est pas essentiellement difficile [...] Il me suffirait d'être immortel pour la mener jusqu'au bout.

Borges [70]

Curriculum Vitae

État-civil et coordonnées

Sébastien Picault, né le 8 avril 1975 ; marié, 3 enfants

LIFL bâtiment M3 bureau 221 (+33 3 59 63 22 37)
ou IUT 'A' bureau 3A/45 (+33 3 59 63 22 37)
Cité Scientifique, 59655 Villeneuve d'Ascq Cedex

Parcours professionnel

depuis 2002

Maître de conférences (classe normale 6e échelon), UFR d'IEEA, Université Lille 1, Villeneuve d'Ascq
Enseignements actuels :
— DUT informatique : Bases de données ;
— licence professionnelle SIL DA2I (Développement et Administration Internet et Intranet) : Génie logiciel ;
— master 2 : Simulation multi-agents
Recherche : au sein du LIFL (UMR CNRS 8022), équipe SMAC (« Systèmes Multi-Agents et Comportements »)

2000–2002

Attaché temporaire d'enseignement et de recherche, Université Paris 6 (UPMC)

1997–2000

Moniteur, Université Paris 13, Villetaneuse

Formation

1997–2001
Doctorat en informatique, Laboratoire d'Informatique de Paris 6 (LIP6, UPMC)
Thèse soutenue le 1er octobre 2001 (directeur : Alexis Drogoul)
Titre : « Modèles de comportements sociaux pour les collectivités de robots et d'agents »
Mots-clefs : simulation multi-agents, robotique collective en milieu ouvert, algorithmes évolutionnistes

1996–1997
DEA d'Intelligence Artificielle (IARFA), Université Paris 6 (UPMC), Paris

1994–1997
Ingénieur télécom, École Nationale Supérieure des Télécommunications, Paris

Activité de recherche

Domaine

Méthodes et outils pour la simulation multi-agents de systèmes complexes
→ simulation orientée interactions, simulation multi-niveaux

Encadrement de thèses

— Adrien Maudet, COGIT, IGN (avec C. Duchêne et G. Touya) — soutenance prévue en 2015
— Yoann Kubera, LIFL, Lille 1 (avec P. Mathieu) — soutenue le 6 décembre 2010
« Simulations orientées-interaction des systèmes complexes »

Jurys de thèses

— 12 sept. 2012 — Thi Thanh Ha Hoang, (LCIS, Valence, Université de Grenoble)
directeur Michel Occello, encadrant Nguyen Thanh Binh (Univ. Danang, Vietnam)
— 23 sept. 2005 — Dong Yue Wang, (IBISC, Université d'Évry)
directrice Pascale Le Gall, encadrant Guillaume Hutzler
— 17 janv. 2003 — Samuel Landau (LIP6, Université Paris 6)
directeur Alexis Drogoul

Encadrements divers

Avec Philippe Mathieu, stages de DEA, Master 2 ou Ingénieur (Lille 1) :
— Guillaume Dauster, 2013 : « INBASU : INdividual BAsed SUpermarket simulation » (projet INBASU)
— Florent Delhaye, 2010 : « Le multi-niveaux pour la simulation multi-agents : application à une simulation de centre commercial » (projet PADAWAN)

— François Gaillard, 2008 : « Rétro Ingénierie pour simulations centrées individus » (projet LEIA)
— Yoann Kubera, 2007 : « La complexité dans les simulations multi-agents » (projet IODA)
— Benoît Lacroix, 2006 : « La gestion du temps et de l'espace dans les simulations de foules » (projet MiSC)
— Laetitia Bonte, 2005 : « Représentation multi-échelles pour les plateformes à grands nombres d'agents » (projet SimuLE)
— Benoît Gosse, 2004 : « Méthodes de simulation à large échelle par agents situés réactifs. Application à la biologie » (projet SimuLE)
— Jérémy Scafi, 2003 : « Projet "PASPAS" : PlAteforme de Simulation Pour Agents Situés » (stage Polytech'Lille)

Avec Cécile Duchêne (IGN), stages de master recherche (Paris XI) :
— Adrien Maudet, 2012 : « Adaptation de la partie comportementale du modèle multi-agents PADAWAN pour la résolution de problème en cartographie »
— Ala Atrash, 2011 : « Adaptation of PADAWAN multi-agent model for the solution of a spatial problem : The cartographic generalization »

Dans le cadre de projets contractualisés :
— Marc-Antoine Dupré (2011–2012, 12 mois), ingénieur contractuel (projet Galaxian)
— David Panzoli (2010–2011, 13 mois), post-doctorant (projet Format-Store)
— Jean-Baptiste Leroy (2010–2011, 16 mois), ingénieur contractuel (projet Format-Store)

Mémoires de fin d'études (Master 2 ou Ingénieur) :
— 2012 : Thomas Clément, Master E-Services (projet IODA/Unity)
— 2006 : Benjamin Kubiak et Valentin Ribeiro Dubois, Polytech'Lille (projet SimuLE)
— 2005 : Virgile Bourse et Mickaël Delacroix, DESS IAGL (projet SimuLE)
— 2005 : Cyril Moreau et Nicolas Petitprez, Polytech'Lille (projet SimuLE)

Comités de programme

— JFSMA (*Journées Francophones sur les Systèmes Multi-Agents*) depuis 2005
— PAAMS'2014 (*International Conference on Practical Applications of Agents and Multi-Agent Systems*)

Relecteur

— Elsevier IST (*Information and Software Technology Journal*)
— Hermès TSI (*Technique et Science Informatiques*)
— Hermès RIA (*Revue d'Intelligence Artificielle*)
— AAMAS'03, AAMAS'10 (*Int. Conf. on Autonomous Agents and Multi-Agent Systems*)
— IAT'11, IAT'12 (*Int. Conf. on Intelligent Agent Technology*)
— PAAMS'13 (*Int. Conf. on Practical Applications of Agents and Multi-Agent Systems*)

Organisation de conférences

— 2013 ; Lille : membre de l'équipe organisatrice de la plateforme IA de l'AFIA, regroupant 6 conférences francophones (dont les JFSMA) et 9 ateliers
— 2003, Lille : membre du comité d'organisation de la conférence MFI (Modèles Formels de l'Interaction)
— 2002, Lille : membre du comité d'organisation des JFIADSMA (Journées Francophones sur l'Intelligence Artificielle Distribuée et les Systèmes Multi-Agents)
— 1998, Paris : membre du comité d'organisation d'*Agents' World*, événement regroupant la conférence internationale ICMAS (*International Conference on Multi-Agent Systems*), 6 workshops internationaux et les compétitions RoboCup et MiroSot

Projets contractuels

— 2012 : contrat de conseil et de formation en IA et techniques de simulation pour jeux vidéo, société BigBrowser, Villeneuve d'Ascq, en partenariat avec le MITI Incubateur Nord-Pas-de-Calais (6 mois)
— 2011 : Galaxian, contrat en partenariat avec l'IRCICA (Institut de Recherche sur les Composants logiciels et matériels pour l'Information et la Communication Avancée – Villeneuve d'Ascq), pour la réalisation d'une plateforme de démonstration des techniques de simulation orientée interactions, scénario de bataille spatiale 3D tournant en continu sur un mur d'image (plateforme de réalité virtuelle PIRVI) (12 mois)
— 2009–2011 : Format-Store, réponse à l'appel à projet *Serious Game* du ministère de l'industrie et de la recherche ; en partenariat avec l'entreprise spécialisée dans les mondes virtuels Idées-3Com et l'école de commerce Enaco (25 mois)

Valorisation de la recherche

— 19 mars 2012 — conférencier invité au séminaire du LIP6 (Laboratoire d'Informatique de Paris 6), Paris : « De la simulation centrée individus à la simulation centrée interactions. La méthode IODA »
— 8 sept. 2011 — conférencier invité au séminaire du CEBC (Centre d'Études Biologiques de Chizé, CNRS) : « Des simulations centrées individus... centrées sur les interactions »
— 19 mai 2010 — conférencier invité au séminaire de l'IXXI (Institut Rhône-Alpin des Systèmes Complexes), Lyon : « La simulation multi-agents orientée interactions. Application aux rythmes circadiens »
— démonstration des méthodes et outils de simulation : projet Galaxian (*IBM scientific award* à PAAMS'2013)
— développement, maintenance et mise à disposition de l'extension IODA pour la plateforme de simulation NetLogo, référencée depuis le site web de NetLogo (`http://ccl.northwestern.edu/netlogo/resources.shtml`, rubrique *Tools*)
— mise à disposition avec Yoann Kubera du bytecode du moteur JEDI (archive *jar*) ainsi que de l'applet de démonstration du moteur

Responsabilités

— Membre du comité de sélection de l'Université du Littoral (2013)
— Membre de la commission de spécialistes de l'Université d'Évry (2005–2008)

Divers

— membre de l'AFIA (Association Française pour l'Intelligence Artificielle)
— membre de l'Union Rationaliste

Publications personnelles

Revues internationales

[1] Mathieu, P., Panzoli, D. et Picault, S. 2013, « Virtual customers in a multiagent training application », dans *Transactions on Edutainment IX*, *Lecture Notes in Computer Science*, vol. 7544, Springer, p. 97–114.

[2] Kubera, Y., Mathieu, P. et Picault, S. 2011, « IODA : An interaction-oriented approach for multi-agent based simulations », *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, vol. 23, n° 3, p. 303–343.

Revues nationales

[3] Mathieu, P. et Picault, S. 2012, « Intérêt de la simulation centrée interactions pour les sciences humaines et sociales », *Revue des Nouvelles Technologies de l'Information (RNTI)*, vol. MASHS 2011/2012 : Modèles et Apprentissage en Sciences Humaines et Sociales, RNTI-SHS-1, p. 15–30. Éditions Hermann.

[4] Gaillard, F., Kubera, Y., Mathieu, P. et Picault, S. 2010, « Une méthode pour naviguer dans l'espace des simulations », *Revue d'Intelligence Artificielle (RIA)*, vol. 24, n° 5, p. 575–599. Hermès.

[5] Kubera, Y., Mathieu, P. et Picault, S. 2008, « Formalisation et implémentation des interactions pour la simulation centrée individu », *RSTI – L'objet*, vol. 14, n° 1-2, p. 9–33. Numéro spécial Architectures Logicielles, Hermès.

Conférences internationales (avec comité de lecture et actes)

[6] Mathieu, P. et Picault, S. 2013, « From real purchase to realistic populations of simulated customers », dans *Proceedings of the 11th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'2013)*, *Lecture Notes in Computer Science*, vol. 7879, édité par Y. Demazeau, T. Ishida, J. M. Corchado et J. Bajo, Springer, p. 216–227.

[7] Maudet, A., Touya, G., Duchêne, C. et Picault, S. 2013, « Improving multi-level interactions modelling in a multi-agent generalisation model : first experiments »,

dans *Proceedings of the 16th ICA Workshop on Generalisation and Map Production*, édité par D. Burghardt, Dresden, Germany.

[8] Mathieu, P., Panzoli, D. et Picault, S. 2011, « Format-Store : a multi-agent based approach to experiential learning », dans *Proceedings of the 3rd International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES'2011)*, édité par F. Liarokapis, IEEE Computer Society Press, p. 120–127.

[9] Mathieu, P. et Picault, S. 2011, « An interaction-oriented model for multi-scale simulation », dans *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'2011)*, édité par T. Walsh, IJCAI/AAAI, p. 332–337.

[10] Mathieu, P., Picault, S. et Kubera, Y. 2010, « An interaction-oriented model of customer behavior for the simulation of supermarkets », dans *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'2010)*, édité par X. J. Huang, A. A. Ghorbani, M.-S. Hacid et T. Yamaguchi, IEEE Computer Society Press, p. 407–410.

[11] Gaillard, F., Kubera, Y., Mathieu, P. et Picault, S. 2009, « A reverse engineering form for multi agent systems », dans *Proceedings of the 9th International Workshop Engineering Societies in the Agents World (ESAW'2008)*, *Lecture Notes in Computer Science*, vol. 5485, édité par A. Artikis, G. Picard et L. Vercouter, Springer, p. 137–153.

[12] Kubera, Y., Mathieu, P. et Picault, S. 2009, « How to avoid biases in reactive simulations », dans *Proceedings of the 7th International Conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'2009)*, *Advances in Intelligent and Soft Computing*, vol. 55, édité par Y. Demazeau, J. Pavón, J. M. Corchado et J. Bajo, Springer, p. 100–109.

[13] Kubera, Y., Mathieu, P. et Picault, S. 2009, « Interaction biases in multi-agent simulations : An experimental study », dans *Proceedings of the 9th International Workshop Engineering Societies in the Agents World (ESAW'2008)*, édité par A. Artikis, G. Picard et L. Vercouter, LNAI, Springer, p. 229–247.

[14] Kubera, Y., Mathieu, P. et Picault, S. 2008, « Interaction-oriented agent simulations : From theory to implementation », dans *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI'2008)*, édité par M. Ghallab, C. Spyropoulos, N. Fakotakis et N. Avouris, IOS Press, p. 383–387.

[15] Kubera, Y., Mathieu, P. et Picault, S. 2008, « Interaction selection ambiguities in multi-agent systems », dans *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'2008)*, édité par C. Zhang, N. Cercone et L. Jain, IEEE Computer Society Press, p. 75–78.

[16] Landau, S., Sigaud, O., Picault, S. et Gérard, P. 2005, « An experimental comparison between ATNoSFERES and ACS », dans *Learning Classifier Systems, International Workshops, IWLCS 2003-2005, Revised Selected Papers*, *Lecture Notes in Computer Science*, vol. 4399, édité par T. Kovacs, X. Llorà, K. Takadama, P. L. Lanzi, W. Stolzmann et S. W. Wilson, Springer, p. 144–160.

[17] Landau, S., Picault, S., Sigaud, O. et Gérard, P. 2003, « Further comparison between ATNoSFERES and XCSM », dans *Learning Classifier Systems, 5th International Workshop, IWLCS 2002, Revised Papers*, *Lecture Notes in Computer Science*, vol. 2661, édité par P. L. Lanzi, W. Stolzmann et S. W. Wilson, Springer, p. 99–117.

[18] Landau, S., Picault, S., Sigaud, O. et Gérard, P. 2002, « A comparison between ATNoSFERES and XCSM », dans *GECCO 2002 : Proceedings of the Genetic and Evolutionary Computation Conference*, édité par W. B. Langdon *et al.*, Morgan Kaufmann, p. 926–933.

[19] Landau, S. et Picault, S. 2001, « Modeling adaptive multi-agent systems inspired by developmental biology », dans *Multi-Agent-Systems and Applications II. 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001, Selected Revised Papers*, *Lecture Notes in Computer Science*, vol. 2322, édité par V. Marík, O. Stepánková, H. Krautwurmowa et M. Luck, Springer, p. 221–232.

[20] Landau, S., Picault, S. et Drogoul, A. 2001, « ATNoSFERES : a model for evolutive agent behaviors », dans *Proceedings of the AISB Symposium on Adaptive Agents and Multi-Agent Systems (AISB'2001)*, édité par D. Kudenko et E. Alonso, The Society for the Study of Artificial Intelligence and the Simulation of Behaviour, York, UK, ISBN 1-902956-17-0, p. 95–99.

[21] Picault, S. et Drogoul, A. 2000, « The MICRoBES project, an experimental approach towards 'Open Collective Robotics' », dans *Proceedings of the 5th International Conference on Distributed Autonomous Robotic Systems (DARS'2000)*, édité par L. E. Parker.

[22] Picault, S. et Drogoul, A. 2000, « Robots as a social species : the MICRoBES project », dans *Socially Intelligent Agents : the Human in the Loop (SIA'2000)*, édité par K. Dautenhahn, n° FS–00–04 dans AAAI Fall Symposium, AAAI Press, ISBN 978-1-57735-127-6, p. 139–141.

[23] Picault, S. et Collinot, A. 1998, « Designing social cognition models for multi-agent systems through simulating primate societies », dans *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)*, édité par Y. Demazeau, IEEE Computer Society Press, p. 238–245.

Conférences francophones (avec comité de lecture et actes)

[24] Mathieu, P. et Picault, S. 2013, « Des données aux agents : la simulation réaliste de populations diversifiées de clients », dans *Actes des 21e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2013)*, édité par S. Hassas et M. Morge, Cépaduès, p. 41–50.

[25] Mathieu, P., Panzoli, D. et Picault, S. 2011, « Serious games et SMA. Application à un supermarché virtuel », dans *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, édité par E. Adam et J.-P. Sansonnet, Cépaduès, p. 181–190.

[26] Picault, S., Mathieu, P. et Kubera, Y. 2010, « PADAWAN, un modèle multi-échelles pour la simulation orientée interactions », dans *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, édité par M. Occello et L. Rejeb, Cépaduès, p. 193–202.

[27] Gaillard, F., Kubera, Y., Mathieu, P. et Picault, S. 2009, « Une forme de rétro ingénierie pour systèmes multi agents : explorer l'espace des simulations », dans *Actes des 17e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2009)*, édité par Z. Guessoum et S. Hassas, Cépaduès, p. 135–144.

[28] Kubera, Y., Mathieu, P. et Picault, S. 2007, « La complexité dans les simulations multi-agents », dans *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, édité par V. Camps et P. Mathieu, Cépaduès, p. 139–148.

[29] Mathieu, P., Picault, S. et Routier, J.-C. 2007, « Donner corps aux interactions », dans *Actes des 4e Journées Francophones sur les Modèles Formels de l'Interaction (MFI'07)*, Université de Paris Dauphine, p. 333–340.

[30] Picault, S., Corellou, F., Schwartz, C. et Bouget, F.-Y. 2007, « Simulation multi-agent de réseaux génétiques : les rythmes circadiens d'*ostreococcus tauri* », dans *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, édité par V. Camps et P. Mathieu, Cépaduès, p. 149–158.

[31] Lacroix, B., Mathieu, P. et Picault, S. 2006, « Une gestion réaliste du temps et de l'espace dans les simulations de foules », dans *Actes des 14e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2006)*, édité par V. Chevrier et M.-P. Huget, Hermès, p. 77–80.

[32] Mathieu, P. et Picault, S. 2006, « Vers une représentation des comportements centrée interactions », dans *Actes du 15e congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA'2006)*, édité par P. Bouthemy et Y. Demazeau, Université de Tours. URL `http://www.tsi.enst.fr/afrif/rfia2006/pdf/065.pdf`.

[33] Mathieu, P., Picault, S. et Routier, J.-C. 2003, « Simulation de comportements pour agents rationnels situés », dans *Actes des 2e Journées Francophones sur les Modèles Formels de l'Interaction (MFI'03)*, édité par A. Herzig, B. Chaïb-Draa et P. Mathieu, Cépaduès, p. 277–282.

[34] Picault, S. et Drogoul, A. 1999, « MICRoBES : vers des collectivités de robots socialement situés », dans *Actes des 7e Journées Francophones Intelligence Artificielle Distribuée et Systèmes Multi-Agents (JFIADSMA'99)*, édité par M.-P. Gleizes et P. Marcenac, Hermès, p. 265–277.

[35] Picault, S. et Collinot, A. 1998, « La socialité : étude, enjeux et applications dans les systèmes multi-agents », dans *Actes des 6e Journées Francophones Intelligence Artificielle Distribuée et Systèmes Multi-Agents (JFIADSMA'98)*, édité par J.-P. Barthès, V. Chevrier et C. Brassac, Hermès, p. 327–340.

Articles courts, démonstrations, posters (conf. int. sélectives)

[36] Mathieu, P. et Picault, S. 2013, « The Galaxian project : A 3d interaction-based animation engine », dans *Proceedings of the 11th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'2013)*, *Lecture Notes in Computer Science*, vol. 7879, édité par Y. Demazeau, T. Ishida, J. M. Corchado et J. Bajo, Springer, p. 312–315. Demonstration paper.

[37] Mathieu, P., Panzoli, D. et Picault, S. 2012, « An immersion into a multi-agent store simulation », dans *Proceedings of the 10th International Conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'2012)*, *Advances in Intelligent and Soft Computing*, vol. 155, édité par Y. Demazeau, J. P. Müller, J. M. Corchado et J. Bajo, Springer, p. 273–276. Demonstration paper.

[38] Mathieu, P., Panzoli, D. et Picault, S. 2012, « Virtual customers in an agent world », dans *Proceedings of the 10th International Conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'2012)*, *Advances in Intelligent and Soft Computing*, vol. 155, édité par Y. Demazeau, J. P. Müller, J. M. Corchado et J. Bajo, Springer, p. 147–152. Short paper.

[39] Kubera, Y., Mathieu, P. et Picault, S. 2010, « Everything can be agent ! », dans *Proceedings of the 9th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'2010)*, édité par W. van der Hoek, G. Kaminka, Y. Lespérance, M. Luck et S. Sen, International Foundation for Autonomous Agents and Multiagent Systems, p. 1547–1548. Extended abstract.

[40] Picault, S. 1998, « A multi-agent simulation of primate social concepts », dans *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI'98)*, édité par H. Prade, John Wiley and Sons, p. 327–328. Young researcher paper.

[41] Bersagol, V., Dessalles, J.-L., Kaplan, F., Marze, J.-C. et Picault, S. 1996, « XMOISE : A logical spreadsheet to elicit didactic knowledge », dans *Computer Aided Learning and Instruction in Science and Engineering, Third International Conference, CALISCE'96*, *Lecture Notes in Computer Science*, vol. 1108, édité par A. Díaz de Ilarra Sanchez et I. Fernández de Castro, Springer, p. 430–432. Poster.

Vulgarisation

[42] Mathieu, P., Picault, S. et Routier, J.-C. 2005, « Les agents intelligents », *Pour La Science*, , n° 332, p. 44–52, ISSN 0153-4092.

[43] Picault, S. 2002, « Des moutons et des robots (P. Arnaud) », *La Recherche*, , n° 350, ISSN 0029-5671. Critique de livre.

Ateliers

[44] Lacroix, B., Mathieu, P. et Picault, S. 2006, « Time and space management in crowd simulation », dans *Proceedings of the European Simulation and Modelling Conference (ESM'2006)*, p. 315–320.

[45] Mathieu, P. et Picault, S. 2005, « Towards an interaction-based design of behaviors », dans *Proceedings of the 3rd European Workshop on Multi-Agent Systems (EUMAS'2005)*, édité par M.-P. Gleizes *et al.*, Koninklijke Vlaamse Academie van Belie voor Wetenschappen en Kunsten, p. 167–178.

[46] Landau, S. et Picault, S. 2003, « Developping agents populations with Ethogenetics », dans *Innovative Concepts for Agent-Based Systems : 1st International Workshop on Radical Agent Concepts (WRAC'2002). Revised Papers*, *Lecture Notes in Computer Science*, vol. 2564, édité par W. Truszkowski, C. Rouff et M. G. Hinchey, Springer.

[47] Picault, S. et Landau, S. 2001, « Ethogenetics : an evolutionary approach to agents organization », dans *Actes du colloque ALCAA (Agents Logiciels, Coopération, Apprentissage et Activité humaine)*.

[48] Picault, S. et Landau, S. 2001, « Ethogenetics and the evolutionary design of agent behaviors », dans *Proceedings of the 5th World Multi-Conference on Systemics,*

Cybernetics and Informatics (SCI'2001), édité par N. Callaos, I. Nunes da Silva et J. Molero.

Mémoires

[49] Landau, S. et Picault, S. 2002, « Stack-based gene expression », Rapport technique 2002/11, LIP6.

[50] Picault, S. 2001, *Modèles de comportements sociaux pour les collectivités de robots et d'agents*, Thèse de doctorat, Université Paris VI. URL `http://tel.archives-ouvertes.fr/tel-00851693`.

[51] Picault, S., Servat, D. et Kaplan, F. 1997, « EDEN : un système évolutif endosémantique », Publication interne, École Nationale Supérieure des Télécommunications. ISSN 0751–1345.

Références bibliographiques

Je ne suis pas assez jeune pour tout savoir.

G. B. Shaw

[52] van der Aalst, W. M. 2011, *Process Mining : Discovery, Conformance and Enhancement of Business Processes*, Springer.

[53] van der Aalst, W. M., van Dongen, B. F., Herbst, J., Maruster, L., Schimm, G. et Weijters, A. J. M. M. 2003, « Workflow mining : A survey of issues and approaches », *Data and Knowledge Engineering*, vol. 47, n° 2, p. 237–267.

[54] Abrami, G. 2004, *Niveaux d'organisation dans la modélisation multi-agent pour la gestion des ressources renouvelables. Application à la mise en œuvre de règles collectives de gestion de l'eau agricole dans la basse-vallée de la Drôme*, Thèse de doctorat, ENGREF.

[55] Agrawal, R. et Srikant, R. 1994, « Fast algorithm for mining association rules », dans *Proceedings of the 20th Conference on Very Large Data Bases (VLDB'94)*, édité par J. G. Bocca, M. Jarke et C. Zaniolo, Morgan Kaufmann, p. 487–499.

[56] An, G. 2008, « Introduction of an agent-based multi-scale modular architecture for dynamic knowledge representation of acute inflammation », *Theoretical Biology and Medical Modelling*, vol. 5, n° 11.

[57] Arthur, W. B., Holland, J. H., LeBaron, B., Palmer, R. et Tayler, P. 1997, « Asset pricing under endogenous expectations in an artificial stock market », *Economic Notes*, vol. 26, p. 297–330.

[58] Axelrod, R. 1997, « Advancing the art of simulation in the social sciences », dans *Simulating Social Phenomena*, *Lecture Notes in Economics and Mathematical Systems*, vol. 456, édité par R. Conte, R. Hegselmann et P. Terna, Springer, p. 21–40.

[59] Bachelard, G. 1934, *Le Nouvel Esprit scientifique*, Presses Universitaires de France.

[60] Baeijs, C., Demazeau, Y. et Alvares, L. 1996, « SIGMA : Application of multi-agent systems to cartographic generalization », dans *Proceedings of the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96)*, *Lecture Notes in Computer Science*, vol. 1038, édité par W. Van de Velde et J. W. Perram, Springer, p. 163–176.

[61] Barutcuoglu, Z., Schapire, R. E. et Troyanskaya, O. G. 2006, « Hierarchical multi-label prediction of gene function », *Bioinformatics*, vol. 22, n° 7, p. 830–836.

[62] Beaudouin, R., Monod, G. et Ginot, V. 2008, « Selecting parameters for calibration via sensitivity analysis : An individual-based model of mosquitofish population dynamics », *Ecological Modelling*, vol. 218, p. 29–48.

[63] Bellotti, F., Berta, R., Gloria, A. D. et Primavera, L. 2009, « Enhancing the educational value of video games », *Computers in Entertainment*, vol. 7, n° 23, p. 1–18.

[64] Belousov, B. 1959, « Periodically acting reaction and its mechanism », *Collection of Abstracts on Radiation Medicine*, vol. 147, p. 145.

[65] Bersini, H. 2012, « UML for ABM », *Journal of Artificial Societies and Social Simulation*, vol. 15, n° 1. URL `http://jasss.soc.surrey.ac.uk/15/1/9.html`.

[66] von Bertalanffy, L. 1993, *Théorie générale des systèmes*, Dunod. Traduit de l'anglais par Jean-Benoit Chabrol (*General System Theory*, 1968).

[67] Boltzmann, L. 1902, *Leçons sur la théorie cinétique des gaz*, Gauthier-Villars. Traduit de l'allemand par A. Gallotti (*Vorlesungen über Gastheorie*, 1896–1898).

[68] Bonabeau, E. 2002, « Agent-based modeling : Methods and techniques for simulating human systems », *Proceedings of the National Academy of Sciences (PNAS)*, vol. 99, n° 3, p. 7280–7287.

[69] Bonneaud, S., Redou, P., Thébault, D. et Chevaillier, P. 2007, « Multi-modélisation agent orientée patterns. Application aux écosystèmes exploités », dans *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, édité par V. Camps et P. Mathieu, Cépaduès, p. 119–128.

[70] Borges, J. L. 1967, *Fictions*, chap. « Pierre Menard, auteur du *Quichotte* », La Croix du Sud, Gallimard. Traduit de l'espagnol par Paul Verdevoye (*Pierre Menard, autor del Quijote*, 1939).

[71] Boriah, S., Chandola, V. et Kumar, V. 2008, « Similarity measures for categorical data : A comparative evaluation », dans *Proceedings of the SIAM International Conference on Data Mining (SDM 2008)*, SIAM, p. 243–254.

[72] Boulier, F. 2006, *Réécriture algébrique dans les systèmes d'équations différentielles polynomiales en vue d'applications dans les Sciences du Vivant*, Mémoire d'habilitation à diriger des recherches, Université Lille 1. URL `http://tel.archives-ouvertes.fr/tel-00137153`.

[73] Boulier, F., Lefranc, M., Lemaire, F., Morant, P.-E. et Ürgüplü, A. 2007, « On proving the absence of oscillations in models of genetic circuits », dans *Proceedings of the 2nd International Conference on Algebraic Biology (AB'2007)*, *Lecture Notes in Computer Science*, vol. 4545, édité par H. Anai, K. Horimoto et T. Kutsia, Springer, p. 66–80.

[74] Bousquet, F. et Le Page, C. 2004, « Multi-agent simulations and ecosystem management : a review », *Ecological Modelling*, vol. 176, n° 3–4, p. 313–332.

[75] Brandouy, O., Mathieu, P. et Veryzhenko, I. 2013, « On the design of agent-based artificial stock markets », dans *Agents and Artificial Intelligence, Communications in Computer and Information Science*, vol. 271, édité par J. Filipe et A. Fred, Springer, p. 350–364.

[76] Breton, L. et Jussien, N. 2004, « Un CSP comme comportement d'agent. Application à la résolution d'équations en physique des milieux granulaires », *Journal Électronique d'Intelligence Artificielle*, vol. 3, n° 26.

[77] Caillou, P. et Gil-Quijano, J. 2012, « Description automatique de dynamiques de groupes dans des simulations à base d'agents », dans *Actes des 20e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2012)*, édité par P. Chevaillier et B. Mermet, Cépaduès, p. 23–32.

[78] Camus, B., Siebert, J., Bourjot, C. et Chevrier, V. 2012, « Modélisation multi-niveaux dans AA4MM », dans *Actes des 20e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2012)*, édité par P. Chevaillier et B. Mermet, Cépaduès, p. 43–52.

[79] Chaib-Draa, B., Mandiau, R. et Millot, P. 1992, « Distributed intelligence. An annotated bibliography », *SIGART Bulletin*, vol. 3, n° 3, p. 20–37.

[80] Chen, D., Theodoropoulos, G. K., Turner, S. J., Cai, W., Minson, R. et Zhang, Y. 2008, « Large scale agent-based simulation on the grid », *Future Generation Computer Systems*, p. 658–671.

[81] Choi, S.-S., Cha, S.-H. et Tappert, C. C. 2010, « A survey of binary similarity and distance measures », *Journal of Systemics, Cybernetics and Informatics*, vol. 8, n° 1, p. 43–48.

[82] Collier, N. et North, M. 2012, « Repast HPC : Platform for large-scale agent-based modeling », dans *Large-scale computing techniques for complex system simulations*, édité par W. Dubitzky, K. Kurowski et B. Schott, Parallel and Distributed Computing, John Wiley and Sons, p. 81–109.

[83] Conte, R. et Castelfranchi, C. 1993, « Norms as mental objects. From normative beliefs to normative goals », dans *From Reaction to Cognition. 5th European Workshop on Modelling Autonomous Agents (MAAMAW'93), Lecture Notes in Computer Science*, vol. 957, édité par C. Castelfranchi et J.-P. Müller, Springer, p. 186–196.

[84] Coquillard, P. et Hill, D. 1997, *Modélisation et simulation d'écosystèmes*, Masson, Paris.

[85] Courties, C., Vaquer, A., Troussellier, M., Lautier, J., Chrétiennot-Dinet, M. J., Neveux, J., Machado, C. et Claustre, H. 1994, « Smallest eukaryotic organism », *Nature*, vol. 370, n° 6487, p. 255.

[86] David, D., Payet, D., Botta, A., Lajoie, G., Manglou, S. et Courdier, R. 2007, « Un couplage de dynamiques comportementales : Le modèle DS pour l'aménagement du territoire », dans *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, édité par V. Camps et P. Mathieu, Cépaduès, p. 129–138.

[87] David, D., Payet, D. et Courdier, R. 2011, « Réification de zones urbaines émergentes dans un modèle simulant l'évolution de la population à La Réunion », dans *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, édité par E. Adam et J.-P. Sansonnet, Cépaduès, p. 63–72.

[88] Demazeau, Y. 1995, « From interactions to collective behaviour in agent-based systems », dans *Proceedings of the 1st European Conference on Cognitive Science*, Saint-Malo, France.

[89] Derelle, E., Ferraz, C., Rombauts, S., Rouzé, P., Worden, A. Z., Robbens, S., Partensky, F., Degroeve, S., Echeynié, S., Cooke, R., Saeys, Y., Wuyts, J., Jabbari, K., Bowler, C., Panaud, O., Piégu, B., Ball, S. G., Ral, J.-P., Bouget, F.-Y., Piganeau, G., De Baets, B., Picard, A., Delseny, M., Demaille, J., Van de Peer, Y. et Moreau, H. 2006, « Genome analysis of the smallest free-living eukaryote *Ostreococcus tauri* unveils many unique features », *Proceedings of the National Academy of Sciences (PNAS)*, vol. 103, n° 31.

[90] Devigne, D., Mathieu, P. et Routier, J.-C. 2004, « Planning for spatially situated agents », dans *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'2004)*, IEEE Computer Society Press, p. 385–388.

[91] Dice, L. R. 1945, « Measures of the amount of ecologic association between species », *Ecology*, vol. 26, n° 3, p. 297–302.

[92] Doran, J. 1995, « Simulating societies using distributed artificial intelligence », dans *Dagstuhl Seminar Social Science Microsimulation*, édité par K. G. Troitzsch, U. Mueller, N. Gilbert et J. Doran, Springer, ISBN 3-540-61572-5, p. 381–393.

[93] Dorigo, M. et Gambardella, L. M. 1997, « Ant colony system : a cooperative learning approach to the traveling salesman problem », *IEEE Transactions on Evolutionary Computation*, vol. 1, n° 1, p. 53–66.

[94] Doyle, A. C. 1890, « The Sign of the Four », *Lippincott's Monthly Magazine*, p. 145–223. February.

[95] Drogoul, A. 1993, *De la Simulation Multi-Agent à la Résolution Collective de Problèmes. Une étude de l'émergence de structures d'organisation dans les Systèmes Multi-Agents*, Thèse de doctorat, Université Paris VI.

[96] Drogoul, A. et Dubreuil, C. 1991, « Eco-problem-solving model : Results of the N-Puzzle », dans *Decentralized AI III*, édité par Y. Demazeau et E. Werner, Elsevier, p. 283–295.

[97] Drogoul, A. et Ferber, J. 1993, « From Tom-Thumb to the Dockers : Some experiments with foraging robots », dans *From Animals to Animats 2. Proceedings of the 2nd International Conference on Simulation of Adaptive Behavior*, édité par J.-A. M. Meyer, H. L. Roitblat et S. W. Wilson, MIT Press, Cambridge, MA, p. 451–459.

[98] Drogoul, A. et Ferber, J. 1994, « Multi-agent simulations as a tool for modeling societies : Application to social differentiation in ant colonies », dans *Artificial Social Systems. 4th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'92)*, *Lecture Notes in Computer Science*, vol. 830, édité par C. Castelfranchi et E. Werner, Springer, p. 3–23.

[99] Drogoul, A., Vanbergue, D. et Meurisse, T. 2003, « Multi-agent based simulation : Where are the agents ? », dans *Proceedings of the 3rd International Conference on Multi-Agent Based Simulation (MABS'2002)*, *Lecture Notes in Computer Science*, vol. 2581, édité par J. S. Sichman, F. Bousquet et P. Davidsson, Springer, p. 1–15.

[100] Duchêne, C. 2004, *Généralisation par agents communicants : Le modèle CARTACOM. Application aux données topographiques en zone rurale*, Thèse de doctorat, Université Paris VI.

[101] Duchêne, C., Ruas, A. et Cambier, C. 2012, « The CartACom model : transforming cartographic features into communicating agents for cartographic generalisation », *International Journal of Geographical Information Science*, vol. 26, n° 9, p. 1533–1562.

[102] Duchêne, C. et Touya, G. 2010, « Emergence de zones conflits dans deux modèles de généralisation cartographique multi-agents », dans *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, édité par M. Occello et L. Rejeb, Cépaduès, p. 33–42.

[103] Dujardin, T. et Routier, J.-C. 2010, « Behavior design of game character's agent », dans *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'2010)*, IEEE Computer Society Press, p. 423–430.

[104] E, W. 2011, *Principles of Multiscale Modeling*, Cambridge University Press.

[105] Edmonds, B. 2005, « Simulation and complexity – how they can relate », dans *Virtual World of Precision. Computer-based Simulations in the Sciences and Social Sciences*, édité par V. Feldmann et K. Mühlfeld, Lit Verlag, p. 5–32.

[106] Edmonds, B. et Hales, D. 2003, « Replication, replication and replication : Some hard lessons from model alignment », *Journal of Artificial Societies and Social Simulations*, vol. 6, n° 4.

[107] Engquist, B., Lötstedt, P. et Runborg, O., (éditeurs). 2009, *Multiscale Modeling and Simulation in Science*, *Lecture Notes in Computational Science and Engineering*, vol. 66, Springer.

[108] Erceau, J. et Ferber, J. 1991, « L'Intelligence Artificielle Distribuée », *La Recherche*, vol. 233.

[109] Espié, S. 1999, « Vehicle driven simulator versus traffic-driven simulator : the INRETS approach », dans *Proceedings of Driving Simulation Conference (DSC'99)*, Paris, France.

[110] Ferber, J. 1995, *Les Systèmes Multi-Agents. Vers une intelligence collective*, InterÉditions.

[111] Ferber, J., Michel, F. et Báez, J. 2005, « AGRE : Integrating environments with organizations », dans *Environments for Multi-Agent Systems : 1st International Workshop (E4MAS'2004). Revised Selected Papers*, *Lecture Notes in Computer Science*, vol. 3374, édité par D. Weyns, H. V. D. Parunak et F. Michel, Springer, p. 48–56.

[112] Fibich, G. et Gibori, R. 2010, « Aggregate diffusion dynamics in agent-based models with a spatial structure », *Operations Research*, vol. 58, n° 5, p. 1450–1468.

[113] François, P. et Hakim, V. 2005, « Core genetic module : the mixed feedback loop », *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 72, n° 031908.

[114] Fujimoto, R. M. 1998, « Time management in the high level architecture », *Simulation*, vol. 71, p. 388–400.

[115] Galán, J. M., Izquierdo, L. R., Izquierdo, S. S., Santos, J. I., del Olmo, R., López-Paredes, A. et Edmonds, B. 2009, « Errors and artefacts in agent-based modelling », *Journal of Artificial Societies and Social Simulation*, vol. 12, n° 1.

[116] Gardner, M. 1970, « Mathematical Games. The fantastic combinations of John Conway's new solitaire game "life" », *Scientific American*, , n° 223, p. 120–123.

[117] Gasser, L. et Briot, J.-P. 1992, « Object-based concurrent programming and Distributed Artificial Intelligence », dans *Distributed Artificial Intelligence : Theory and Praxis*, édité par N. M. Avouris et L. Gasser, Kluwer, p. 81–107.

[118] Gaud, N. 2007, *Systèmes multi-agents holoniques : de l'analyse à l'implantation. Méta-modèle, méthodologie, et simulation multi-niveaux*, Thèse de doctorat, Université de Franche-Comté et Univesité de Technologie de Belford-Montbéliard.

[119] Gibson, J. J. 1979, *The Ecological Approach to Visual Perception*, Hillsdale.

[120] Gil-Quijano, J. 2007, *Modèles d'auto-organisation pour l'émergence de formes urbaines à partir de comportements individuels à Bogota*, Thèse de doctorat, Université Paris VI.

[121] Gilbert, N. et Conte, R. 1995, *Artificial Societies*, UCL Press, London.

[122] Goldbeter, A. 1995, « A model for circadian oscillations in the *Drosophila* period protein (PER) », *Proceedings of the Royal Society of London. Series B, Biological sciences*, vol. 261, p. 319.

[123] Goldbeter, A. 1996, *Biochemical Oscillations and Cellular Rhythms : The molecular bases of periodic and chaotic behaviour*, Cambridge University Press.

[124] Gonze, D., Halloy, J. et Goldbeter, A. 2002, « Deterministic versus stochastic models for circadian rhythms », *Journal of Biological Physics*, vol. 28, p. 637–653.

[125] Grevisse, M. 1993, *Le bon usage. Grammaire française*, 13e éd., Duculot. Refondue par André Goose. 4e tirage (1997).

[126] Harmer, S., Panda, S. et Kay, S. 2001, « Molecular bases of circadian rhythms », *Annual Review of Cell and Developmental Biology*, vol. 17, p. 215–253.

[127] Hayes-Roth, B. 1985, « A blackboard architecture for control », *Artificial Intelligence*, vol. 26, n° 3, p. 251–321.

[128] Hewitt, C. 1977, « Viewing control structures as patterns of message passing », *Artificial Intelligence*, vol. 8, n° 3, p. 323–374.

[129] Hewitt, C. et Inman, J. 1991, « Distributed artificial intelligence betwixt and between : From 'intelligent agents' to open system science », *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, n° 6, p. 1409–1419.

[130] Hiebeler, D. E. 1994, « The Swarm simulation system and Individual-Based Modeling », Working paper 94-11-065, Santa Fe Institute.

[131] Hofstadter, D. 1988, *Ma Thémagie : En quête de l'essence de l'esprit et du sens*, chap. 10 (Des variations sur un thème considérées comme l'essence de la créativité), InterÉditions, Paris, p. 220–248. Traduit de l'anglais par Jean-Luc Bonnetain (*Metamagical Themas : Questing for the Essence of Mind and Pattern*, 1985).

[132] Hutzler, G. et Gortais, B. 2004, « From computer art to ambient displays », *Machine Graphics and Vision*, vol. 13, p. 181–191.

[133] Jaccard, P. 1901, « Étude comparative de la distribution florale dans une portion des Alpes et du Jura », *Bulletin de la Société Vaudoise des Sciences Naturelles*, vol. 37, p. 547–579.

[134] Kubera, Y. 2010, *Simulations orientées-interaction des systèmes complexes*, thèse de doctorat, Université Lille 1.

[135] Kulczynski, S. 1927, « Classe de sciences mathématiques et naturelles », *Bulletin International de l'Académie Polonaise des Sciences et des Lettres*, vol. Série B (Sciences Naturelles), n° Supplément II, p. 57–203.

[136] Lacroix, B. 2009, *Normer pour mieux varier ? La différenciation comportementale par les normes, et son application au trafic dans les simulateurs de conduite*, Thèse de doctorat, Université Lille 1.

[137] Lacroix, B., Mathieu, P. et Kemeny, A. 2013, « Formalizing the construction of populations in multi-agent simulations », *Journal of Engineering Applications of Artificial Intelligence*, vol. 26, n° 1, p. 211–226.

[138] Lamarche-Perrin, R., Demazeau, Y. et Vincent, J.-M. 2011, « Observation macroscopique et émergence dans les SMA de très grande taille », dans *Actes des 19e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2011)*, édité par E. Adam et J.-P. Sansonnet, Cépaduès, p. 53–62.

[139] Lamy, S., Ruas, A., Demazeau, Y., Jackson, M., Mackaness, W. et Weibel, R. « The applications of agents in automated map generalisation », p. 1225–1234.

[140] Langfelder, P. et Horvath, S. 2012, « Fast R functions for robust correlations and hierarchical clustering », *Journal of Statistical Software*, vol. 46, n° 11, p. 1–17.

[141] Langton, C. 1988, *Artificial Life*, Addison-Wesley.

[142] Leloup, J.-C., Gonze, D. et Goldbeter, A. 1999, « Limit cycle models for circadian rhythms based on transcriptional regulation in *Neurospora* and *Drosophila* », *Journal of Biological Rhythms*, vol. 14, p. 433–448.

[143] Lewis, J. 2003, « Autoinhibition with transcriptional delay : a simple mechanism for the zebrafish somitogenesis oscillator », *Current Biology*, vol. 13, p. 1398.

[144] Littré, P.-E. 1863–1872, *Dictionnaire de la langue française*, Hachette.

[145] Locke, J. C. W., Millar, A. J. et Turner, M. S. 2005, « Modelling genetic networks with noisy and varied experimental data : the circadian clock in *arabidopsis thaliana* », *Journal of Theoretical Biology*, vol. 234, p. 383.

[146] Lorenz, K. 1984, *Les fondements de l'éthologie*, Flammarion. Traduit de l'allemand par Jeanne Etoré (*Vergleichende Verhaltensforschung : Grundlagen der Ethologie*, 1978).

[147] Maloney, J., Resnick, M., Rusk, N., Silverman, B. et Eastmond, E. 2010, « The scratch programming language and environment », *ACM Transactions on Computing Education*, vol. 10, n° 4, p. 16 :1–16 :15.

[148] Marcenac, P. et Giroux, S. 1998, « GEAMAS : A generic architecture for agent-oriented simulations of complex processes », *Applied Intelligence*, vol. 8, n° 3, p. 247–267.

[149] Marsh, S. 1994, « Trust in Distributed Artificial Intelligence », dans *Artificial Social Systems. 4th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'92)*, *Lecture Notes in Computer Science*, vol. 830, édité par C. Castelfranchi et E. Werner, Springer, p. 94–112.

[150] Mathieu, P., Routier, J.-C. et Urro, P. 2001, « Un modèle de simulation agent basé sur les interactions », dans *Actes des Premières Journées Francophones sur les Modèles Formels de l'Interaction (MFI'01)*, p. 407–417.

[151] Mathieu, P. et Secq, Y. 2012, « Environment updating and agent scheduling policies in agent-based simulators », dans *Proceedings of the 4th International Conference on Agents and Artificial Intelligence (ICAART'2012)*, édité par J. Filipe et A. L. N. Fred, SciTePress, p. 170–175.

[152] McFarland, D., (éditeur). 1990, *Dictionnaire du comportement animal*, Bouquins, Robert Laffont, Paris. Traduit de l'anglais par Guy Schœller (*The Oxford Companion to Animal Behaviour*, 1981).

[153] Michaelis, L. et Menten, M. L. 1913, « Die Kinetik der Invertinwirkung », *Biochemische Zeitschrift*, vol. 49, p. 333–369.

[154] Michel, F. 2007, « Le modèle IRM4S. De l'utilisation des notions d'influence et de réaction pour la simulation de systèmes multi-agents », *Revue d'Intelligence Artificielle*, vol. 21, n° 5–6, p. 757–779.

[155] Michel, F. 2013, « Intégration du calcul sur GPU dans la plate-forme de simulation multi-agent générique TurtleKit3 », dans *Actes des 21e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2013)*, édité par S. Hassas et M. Morge, Cépaduès, p. 189–198.

[156] Michel, F., Ferber, J., Laur, P.-A. et Aleman, F. 2010, « Programmation situationnelle : programmation visuelle de comportements agents pour non informaticiens », dans *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, édité par M. Occello et L. Rejeb, Cépaduès, p. 65–74.

[157] Michel, F., Gouaich, A. et Ferber, J. 2003, « Weak interaction and strong interaction in agent based simulations », dans *Multi-Agent-Based Simulation III. 4th International Workshop (MABS'2003)*, *Lecture Notes in Computer Science*, vol. 2927, édité par D. Hales, B. Edmonds, E. Norling et J. Rouchier, Springer, p. 43–56.

[158] Minar, N., Burkhart, R., Langton, C. et Askenazi, M. 1996, « The SWARM simulation system : a toolkit for building multi-agent simulations », Working Paper 96-06-042, Santa Fe Institute. URL `http://www.cs.uga.edu/~maria/pads/papers/swarm-MinarEtAl96.pdf`.

[159] Pic de la Mirandole, J. 2005, *De la dignité de l'homme*, 4e éd., Collection « philosophie imaginaire », Éditions de l'éclat. Traduit du latin par Yves Hersant (*Oratio de hominis dignitate*, 1486).

[160] Moncion, T., Amar, P. et Hutzler, G. 2010, « Automatic characterization of emergent phenomena in complex systems », *Journal of Biological Physics and Chemistry*, vol. 10, n° 1, p. 16–23.

[161] Monod, J. 1970, *Le Hasard et la Nécessité. Essai sur la philosophie naturelle de la biologie moderne*, Seuil, Paris.

[162] Morant, P.-E., Vandermoere, C., Parent, B., Lemaire, F., Corellou, F., Schwartz, C., Bouget, F.-Y. et Lefranc, M. 2007, « Oscillateurs génétiques simples. applications à l'horloge circadienne d'une algue unicellulaire », dans *Actes des Rencontres du non linéaire*, Paris. URL `http://nonlineaire.univ-lille1.fr`.

[163] Morvan, G. 2013, « Multi-level agent-based modeling – bibliography », Work in progres, CoRR. URL `http://arxiv.org/abs/1205.0561`, arXiv :1205.0561 [cs.MA].

[164] Morvan, G., Veremme, A. et Dupont, D. 2010, « IRM4MLS : The Influence Reaction model for multi-level simulation », dans *Multi-Agent-Based Simulation XI. International Workshop (MABS'2010)*, *Lecture Notes in Computer Science*, vol. 6532, édité par T. Bosse, A. Geller et C. M. Jonker, Springer, p. 16–27.

[165] Narain, R., Golas, A., Curtis, S. et Lin, M. C. 2009, « Aggregate dynamics for dense crowd simulation », *ACM Transactions on Graphics*, vol. 28, n° 5, p. 122 :1–122 :8.

[166] North, M. J., Collier, N. T. et Vos, J. R. 2006, « Experiences creating three implementations of the Repast agent modeling toolkit », *ACM Transactions on Modeling and Computer Simulations*, vol. 16, n° 1, p. 1–25.

[167] Ochiai, A. 1957, « Zoogeographic studies on the soleoid fishes found in Japan and its neighbouring regions », *Bulletin of the Japanese Society for Fish Science*, vol. 22, p. 526–530.

[168] Omicini, A., Ricci, A. et Viroli, M. 2008, « Artifacts in the A& A meta-model for multi-agent systems », *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, vol. 17, n° 3, p. 435–456.

[169] Panzoli, D., Peters, C., Dunwell, I., Sanchez, S., Petridis, P., Protopsaltis, A., Scesa, V. et de Freitas, S. 2010, « Levels of interaction : A user-guided experience in large-scale virtual environments », *Proceedings of the 2nd International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES'2010)*, p. 87–90.

[170] Papert, S. 1981, *Jaillissement de l'esprit. Ordinateurs et apprentissage*, Flammarion. Traduit de l'anglais par Rose-Marie Vassallo-Villaneau (*Mindstorms : Children, Computers, and Powerful Ideas*, 1980).

[171] Păun, G. et Rozenberg, G. 2002, « A guide to membrane computing », *Theoretical Computer Science*, vol. 287, n° 1, p. 73–100.

[172] Payet, D., Courdier, R., Ralambondrainy, T. et Sébastien, N. 2006, « Le modèle à Temporalité : pour un équilibre entre adéquation et optimisation du temps dans les simulations agent », dans *Actes des 14e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2006)*, édité par V. Chevrier et M.-P. Huget, Hermès, p. 63–76.

[173] Perret, J., Curie, F., Gaffuri, J. et Ruas, A. 2010, « Un système multi-agent pour la simulation des dynamiques urbaines », dans *Actes des 18e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2010)*, édité par M. Occello et L. Rejeb, Cépaduès, p. 205–213.

[174] Pitrat, J. 1990, *Métaconnaissance, futur de l'Intelligence Artificielle*, Hermès, Paris.

[175] Poincaré, H. 1902, *La science et l'hypothèse*, Ernest Flammarion.

[176] Popper, K. R. 2006, *Conjectures et réfutations : la croissance du savoir scientifique*, Bibliothèque scientifique, Payot. Traduit de l'anglais par Michelle Irène et Marc B. de Launay (*Conjectures and Refutations : The Growth of Scientific Knowledge*, 1963).

[177] Quéau, P. 1986, *Éloge de la simulation. De la vie des langages à la synthèse des images*, Champ Vallon/INA.

[178] Rao, A. S. et Georgeff, M. P. 1991, « Modelling rational agents within a BDI-architecture », dans *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, édité par J. Allen, R. Fikes et E. Sandewall, Morgan Kaufmann Publishers, San Mateo, CA.

[179] Resnick, M. 1997, *Turtles, Termites and Traffic Jams*, MIT Press.

[180] Reynolds, C. W. 1987, « Flocks, herds and schools : A distributed behavioral model », *Computer Graphics*, vol. 21, n° 4, p. July.

[181] Roberts, S. C. et Lee, J. D. 2012, « Using agent-based modeling to predict the diffusion of safe teenage driving behavior through an online social network », dans *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 56, p. 2271–2275.

[182] Robinson, S. 2009, « Conceptual modelling for simulation part I : Definition and requirements », *Journal of the Operational Research Society*, vol. 59, n° 3, p. 278–290.

[183] Rodin, V., Querrec, G., Ballet, P., Bataille, F.-R., Desmeulles, G. et Tisseau, J. 2009, « Multi-agents system to model cell signalling by using fuzzy cognitive maps. Application to computer simulation of multiple myeloma », dans *Proceedings of the 9th IEEE International Conference on Bioinformatics and Bioengineering (BIBE'2009)*, édité par J. J. P. Tsai, P. C.-Y. Sheu et H. C. W. Hsia, IEEE Computer Society Press, p. 236–241.

[184] Rouchier, J. et Bousquet, F. 1998, « Non-merchant economy and multi-agent system : An analysis of structuring exchanges », dans *Multi-Agent Systems and Agent-Based Simulation, First International Workshop, MABS '98, Lecture Notes in Computer Science*, vol. 1534, édité par J. S. Sichman, R. Conte et N. Gilbert, Springer, p. 111–123.

[185] Rousseeuw, P. J. 1987, « Silhouettes : a graphical aid to the interpretation and validation of cluster analysis », *Journal of Computational and Applied Mathematics*, vol. 20, p. 53–65.

[186] Routier, J.-C. 2005, *Conception par agent orientée compétences*, Mémoire d'habilitation à diriger des recherches, Université Lille 1.

[187] Routier, J.-C., Mathieu, P. et Secq, Y. 2001, « Dynamic skill learning : A support to agent evolution », dans *Proceedings of the Artificial Intelligence and the Simulation of Behaviour symposium on Adaptive Agents and Multi-agent systems (AISB'2001)*, édité par D. Kudenko et E. Alonso, p. 25–32.

[188] Ruas, A. 1999, *Modèle de généralisation de données géographiques à base de contraintes et d'autonomie*, Thèse de doctorat, Université Marne la Vallée.

[189] Russell, S. J. et Norvig, P. 1995, *Artificial Intelligence. A Modern Approach*, Prentice Hall.

[190] Salome, P. et McClung, C. 2004, « The *Arabidopsis thaliana* clock », *Journal of Biological Rhythms*, vol. 19, p. 425–435.

[191] Sargent, R. G. « Verification and validation of simulation models », dans *Proceedings of the 30th Winter Simulation Conference (WSC'98)*, IEEE Computer Society Press, Los Alamitos, CA, p. 121–130.

[192] Scheiner-Bobis, G. 2002, « The sodium pump. Its molecular properties and mechanisms of ion transport », *European Journal of Biochemistry*, vol. 269, n° 10, p. 2424–2433.

[193] Schelling, T. C. 1971, « Dynamic models of segregation », *Journal of Mathematical Sociology*, vol. 1, p. 143–186.

[194] Schwaiger, A. et Stahmer, B. 2003, « SimMarket : Multiagent-based customer simulation and decision support for category management », dans *Multiagent System Technologies. Proceedings of MATES 2003*, *Lecture Notes in Computer Science*, vol. 2831, édité par M. Schillo, M. Klusch, J. Müller et H. Tianfield, Springer, p. 74–84.

[195] Servat, D. 2000, *Modélisation de dynamiques de flux par agents. Application aux processus de ruissellement, infiltration, et érosion*, Thèse de doctorat, Université Paris VI.

[196] Shao, W. et Terzopoulos, D. 2007, « Autonomous pedestrians », *Graphical Models*, vol. 69, n° 5–6, p. 246–274.

[197] Siebers, P.-O., Aickelin, U., Celia, H. et Clegg, C. W. 2007, « Using intelligent agents to understand management practices and retail productivity », dans *Proceedings of the Winter Simulation Conference (WSC'07)*, Washington, D.C., p. 2212–2220.

[198] Simon, H. A. 1969, *The Science of the Artificial*, MIT Press, Cambridge, MA.

[199] Soler, L. 2000, *Introduction à l'épistémologie*, Ellipses.

[200] Sundman, K. F. 1913, « Mémoire sur le problème des trois corps », *Acta Mathematica*, vol. 36, p. 105–179.

[201] Sycara, K. P. 1998, « Multiagent systems », *AI Magazine*, vol. 19, n° 2, p. 79–92.

[202] Sørensen, T. 1948, « A method of establishing groups of equal amplitude in plant sociology based on similarity of species content », *Kongelige Danske Videnskabernes Selskab. Biol. krifter*, vol. Bd. V, n° 4, p. 1–34.

[203] Taillandier, P., Vo, D.-A., Amouroux, E. et Drogoul, A. 2010, « GAMA : A simulation platform that integrates geographical information data, agent-based

modeling and multi-scale control », dans *Principles and Practice of Multi-Agent Systems. 13th International Conference (PRIMA'2010)*, *Lecture Notes in Computer Science*, vol. 7057, édité par N. Desai, A. Liu et M. Winikoff, Springer, p. 242–258.

[204] Terano, T. 2006, « Exploring the vast parameter space of multi-agent based simulation », dans *Multi-Agent-Based Simulation VII, International Workshop (MABS'2006)*, *Lecture Notes in Computer Science*, vol. 4442, édité par L. Antunes et K. Takadama, Springer.

[205] Theraulaz, G. 1991, *Morphogenèse et Auto-organisation des comportements dans les colonies de guêpes polistes Polistes dominilus (Christ)*, Thèse de troisième cycle, Université d'Aix-Marseille.

[206] Thom, R. 1988, *Esquisse d'une sémiophysique. Physique aristotélicienne et théorie des catastrophes*, InterÉditions.

[207] Torrel, J.-C., Lattaud, C. et Heudin, J.-C. 2009, « Complex systems in cosmology : "The Antennae" case study », *Complex*, vol. 2, p. 1887–1897.

[208] Touya, G. et Duchêne, C. 2011, « CollaGen : Collaboration between automatic cartographic generalisation processes », dans *Advances in Cartography and GIScience*, *Lecture Notes in Geoinformation and Cartography*, vol. 1, édité par A. Ruas, Springer, p. 541–558.

[209] Tranouez, P. 2005, *Contribution à la modélisation et à la prise en compte informatique de niveaux de descriptions multiples. Application aux écosystèmes aquatiques*, Thèse de doctorat, Université du Havre.

[210] Troisi, A., Wong, V. et Ratner, M. A. 2005, « An agent-based approach for modeling molecular self-organization », *Proceedings of the National Academy of Sciences (PNAS)*, vol. 102, n° 2, p. 255–260.

[211] Tsoumakas, G. et Katakis, I. 2007, « Multi label classification : An overview », *International Journal of Data Warehousing and Mining*, vol. 3, n° 3, p. 1–13.

[212] Turing, A. 1950, « Computing machinery and intelligence », *Mind*, vol. LXI, n° 236, p. 433–460.

[213] Tyrrell, T. 1993, « The use of hierarchies for action selection », *Adaptive Behavior*, vol. 1, n° 4, p. 387–420.

[214] Wiener, N. 1948, *Cybernetics : or Control and Communication in the Animal and the Machine*, Cambridge University Press.

[215] Wilde, O. 1890, « The Picture of Dorian Gray », *Lippincott's Monthly Magazine*, p. 1–100. July.

[216] Wilensky, U. 1997, « Netlogo ants model », Netlogo model library, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. URL `http://ccl.northwestern.edu/netlogo/models/Ants`.

[217] Wilensky, U. 1999, « NetLogo », plateforme de simulation, Center for Connected Learning and Computer-Based Modeling (CCL), Northwestern University, Evanston, IL. URL `http://ccl.northwestern.edu/netlogo`.

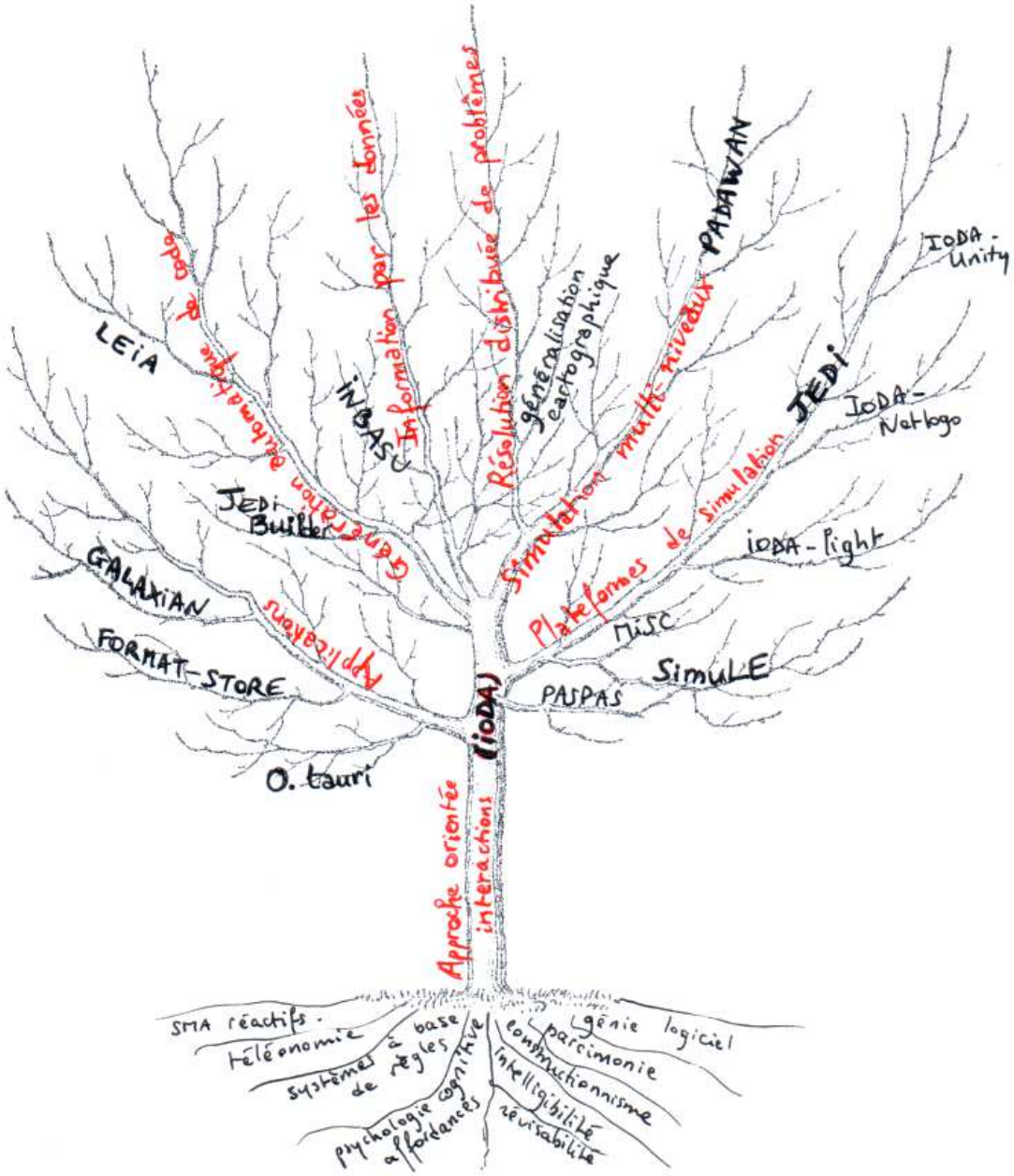
[218] Woo, M.-J., Briot, J.-P. et Ferber, J. 1998, « Using components for modeling intelligent and collaborative mobile agents », dans *Proceedings of the 7th Workshop on Enabling Technologies (WETICE'98)*, IEEE Computer Society Press, p. 276–281.

[219] Zeigler, B. P., Praehofer, H. et Kim, T. G. 2000, *Theory of Modeling and Simulation*, 2e éd., Academic Press, Orlando, FL.

[220] Zhabotinsky, A. 1964, « Periodical process of oxydation of malonic acid solution », *Biophysics*, vol. 9, p. 306–311.

Index général

Ce mémoire d'Habilitation à Diriger des Recherches, rédigé sous théobromine entre Salamanque, La Chapelle sur Erdre, Lille, Villeneuve d'Ascq, Quiberon & autres lieux propices à la réflexion, a été édité sous GNU Emacs et mis en forme sous LaTeX2ε, dans les polices Palatino & Avant-garde.



Villeneuve d'Ascq, 2013

Résumé

Dans ce mémoire de synthèse d'habilitation à diriger des recherches, je présente les travaux en simulation multi-agents que j'ai menés au sein de l'équipe SMAC (LIFL, Université Lille 1) depuis le début de ma carrière d'enseignant-chercheur en 2002. Ceux-ci portent sur la conception et la mise en application de méthodes et d'outils de simulation destinés à faciliter la modélisation de systèmes complexes à large échelle.

Dans ce but, j'ai développé avec mes collègues une approche « orientée interactions » caractérisée par une unification des concepts utilisés dans le domaine des SMA. Elle a donné lieu à une importante élaboration méthodologique et algorithmique (la méthode IODA) dans laquelle toute entité du modèle est représentée par un agent, et tout comportement par une règle appelée interaction. Cette méthode s'appuie sur une séparation entre déclaratif et procédural qui facilite l'acquisition de l'expertise auprès des thématiciens. Par ailleurs de nombreux outils logiciels sont nés de ces recherches (dont la plateforme JEDI et une extension IODA pour la plateforme NetLogo), ainsi que diverses applications dans des domaines variés (biologie cellulaire, *serious games*, marketing, cartographie).

Pour conclure, je présente mon projet de recherche pour les prochaines années qui se propose d'articuler des problématiques issues de travaux récents, d'une part sur la simulation multi-niveaux (qui vise à définir un cadre opérationnel permettant le changement d'échelle d'observation ou de point de vue sur les sous-systèmes d'un système complexe), et d'autre part sur la recherche automatique d'informations dans des données réelles pour augmenter le réalisme comportemental des populations d'agents. Par ailleurs une collaboration avec l'IGN sur l'utilisation de ces techniques pour la généralisation cartographique permet également d'envisager la transposition de ces méthodes de simulation à la résolution de problèmes.

Abstract

In this Habilitation Thesis, I present my research activity in the field of multiagent simulation within the SMAC team (LIFL, Université Lille 1), since the beginning of my career of teacher and researcher in 2002. My work focuses on the design and implementation of methods and tools aimed at facilitating the modeling of large-scale complex systems.

Therefore, I developped with my colleagues an « interaction-oriented » approach, characterized by the unification of several concepts used in the field of MAS. It is supported by a large methodological and algorithmic elaboration (the IODA method), in which each entity of the model is represented by an agent and each behavior by a rule called an interaction. This method relies upon the separation between declarative and procedural features, which facilitates the acquisition of thematic expertise. In addition, several software tools have been built throughout this research (e.g. the JEDI platform and the IODA exension for the NetLogo platform). This method has also been applied to various fields (cell biology, serious games, marketing, cartography).

To conclude, I present my research project for the coming years, which proposes to combine issues raised by recent work : on the one hand, the multi-level simulation (aimed at defining an operational framework for automatizing the change of observation scale or point of view upon the subsystems of a complex phenomenon) ; and on the other hand, the automatic information retrieval from real data, in order to increase the behavioral realism of populations of agents. In addition, a collaboration with the IGN on using those techniques for cartographic generalization opens perspectives towards the transposition of those simulation methods to the field of problem solving.