Thèse de doctorat de l'Université de Paris VI
Université Pierre et Marie Curie

Spécialité informatique

Jean-Charles Campagne

pour obtenir le titre de Docteur de l'Université de Paris VI

Systèmes multi-agents et morphologie

26 septembre 2005

Composition du jury :

Directeur de thèse :	Pr. Alain Cardon
Rapporteur :	Pr. Jacques Ferber
Rapporteur :	Pr. Jean-Paul Sansonnet
Examinateur :	Pr. Amal El Fallah Seghrouchni
Examinateur :	Pr. Paul Bourgine
Examinateur :	Pr. Jean-Pierre Briot

Remerciements

Je tiens à remercier mon directeur de thèse, Alain Cardon, pour avoir eu l'opportunité de travailler sur un sujet si passionnant, et pour tous les échanges tous aussi intéressants que j'ai pu avoir lors de mon doctorat au sein du Laboratoire d'Informatique de Paris 6.

Merci aux membres du jury pour leur évaluation de mon travail de thèse, en particulier mes deux rapporteurs Jacques Ferber et Jean-Paul Sansonnet. Merci également à mes examinateurs Amal Amal El Fallah Seghrouchni, Paul Bourgine et Jean-Pierre Briot.

Merci à toutes les personnes que j'ai rencontrées pendant mon séjour au Laboratoire d'Informatique de Paris 6, aux thésards avec qui j'ai partagé le bureau : Mickael Camus, Paul Guyot, David Julien, Hugues Lecarpentier, Leila Merghem, Minh Nguyen, Valérie Renault.

Merci à mes amis qui m'ont entouré et encouragé.

Je tiens à remercier chaleureusement ma famille, tout particulièrement mes parents, mes grands-parents, mon frère et ma sœur qui m'ont toujours soutenu tout au long de la thèse et des périodes difficiles. Merci à mes proches en Allemagne en Belgique, en Espagne, aux États-Unis, en France, au Luxembourg, à Tahiti, au Royaume-Uni et en Russie pour leur soutien.

À mon père.

Table des matières

Table des matières **xiii**

Introduction **xix**
Présentation xix
Thèse défendue xxii
Plan du manuscrit xxiii

I État de l'art **1**

1 Systèmes multi-agents **3**
1.1 Notions d'agent et de système multi-agent 3
1.1.1 Agent 3
1.1.2 Système multi-agent 4
1.2 Méthodologie 5
1.2.1 Approches par conception 5
Gaia 7
MaSE et agentTool 9
MESSAGE/UML 11
Tropos 14
Prometheus/PDT 15
Voyelles 17
MADKit/Aalaadin 19
AMAS/Adelfe 23
ZEUS 25
1.2.2 Approches par analyse, observation 27
Méthodes d'analyse formelle 27
Méthodes fondées sur l'observation 28
1.3 Question du contrôle des SMA 30

2 Morphologie **33**
2.1 Notion de morphologie et stabilité structurelle 33
2.2 Exemples 35
2.2.1 En Mathématiques 35
2.2.2 En biologie 36
D'Arcy Thompson 36
Morphométrie 37
Exemple modèle proie-prédateur 39
2.2.3 En physique 43
Exemple du pendule 43
Exemple diagramme d'état de l'eau 44
Contrôle de systèmes chaotiques 46
2.2.4 En informatique 51
Attracteurs de générateurs de nombres aléatoires 51
Dans les systèmes multi-agents 55
Travaux d'Alain Cardon 55
Thèse de Franck Lesage 55
2.3 Remarques 61

II Proposition **63**

3 Approche et modèle proposés **65**
3.1 Introduction 66
3.1.1 Cadre et buts 66
3.1.2 SMA auto-adaptatifs vus par la morphologie 67
3.1.3 Trajectoire libre d'un système 70
3.1.4 Modifications de trajectoires 73
3.2 Hypothèse fondamentale 74
3.3 Organisation aspectuelle 76
3.4 Morphologie 78
3.4.1 Notion de projection 78
3.4.2 Propositions de type de formes 79
Retour sur agents 80
Métrique, équivalence 81
3.5 Analyse du système 82
3.5.1 Apprentissage, construction du modèle 82
3.5.2 Cinématique et dynamique 82
Cinématique 83
Dynamique 85
3.5.3 Graphe d'évolution : le modèle 86

3.6 Contrôle du système 89
3.6.1 Hypothèse du contrôle 89
3.6.2 Mécanismes de contrôle 90
3.7 Extensions 92
3.7.1 Agentification de la morphologie et de l'analyse 92
3.7.2 Sondage morphologique 93

4 Contrôle morphologique d'un SMA réactif 95
4.1 But de l'exemple et description générale 95
4.2 Organisation d'agents 96
4.2.1 Structure des agents 96
4.2.2 Environnement 97
4.2.3 Énergie 98
4.2.4 Création initiale des agents 98
4.2.5 Mort d'un agent 98
4.2.6 Clonage 99
4.2.7 Échange d'énergie 99
4.2.8 Mise à jour de l'« ardeur » 99
4.2.9 Boucle de simulation 100
4.2.10 But : convergence, maintien du niveau de la population . . 100
4.2.11 Organigramme du comportement de l'agent 100
4.3 Description de la méthode d'analyse et de contrôle 103
4.3.1 Choix de la mesure sur les agents 103
4.3.2 Choix de la forme 103
4.3.3 Comparaison de formes 105
Distance 105
Formes équivalentes 105
4.3.4 « Réciproque » 105
4.3.5 Corrélation de formes et de comportement du système . . 107
Classification élémentaire 107
Classification plus fine 108
4.3.6 Apprentissage et retour sur le système 109
4.4 Expériences et résultats 110
4.4.1 Test de l'hypothèse 111
4.4.2 Quelques résultats 112
4.4.3 Étude des paramètres 114
Étude de la variation de la densité 115
Étude de la variation de la marge 118
Étude de la variation de la taille 121
Remarques 124
4.4.4 Formes apprises 124

4.5 Conclusion 126

5 Agentification aspectuelle et prototype 127
5.1 Introduction 127
5.2 Description générale du prototype 129
5.2.1 Éléments du système 129
Composants 130
Organisations 130
Agents 131
5.2.2 Identification des éléments du système : les adresses . . . 131
Système d'adressage 131
Adresses valides 132
5.2.3 Communication entre les éléments : les messages 133
Structure d'un message 133
Types de messages 134
Envois valides de messages 134
5.2.4 Objets 136
Description des objets 136
Composition d'objets 137
Fonctions d'agrégation d'objets et listes d'objets 139
Agrégation d'objets 139
Liste d'objets : agrégation par défaut 139
Agrégation d'un objet et d'une liste d'objets 139
Agrégation de deux listes d'objets 139
Difficultés de la réalisation de l'agrégation 140
Métrique sur l'espace d'objets 140
5.2.5 Agents aspectuels 141
Notion de rôle de l'agent aspectuel 142
Interactions entre les agents aspectuels 143
Agents aspectuels de type groupe 144
Notion de rôle d'un groupe 144
5.3 Comportement des agents simples et des agents groupes 145
5.3.1 Agent simple 145
État de veille de l'agent aspectuel simple 146
État de mort d'un agent 147
État d'activation de l'agent aspectuel simple 147
Évaluation du contexte de l'agent aspectuel simple 148
Bilan énergétique 148
Activation du rôle 149
Mise à jour des préférences 149
Activation des règles de comportement 150

5.3.2 Agent aspectuel groupe 152
Rôle agrégateur de l'agent groupe 153
Cycle de vie d'un groupe 154
Création d'un groupe 154
Intégration dans un groupe existant 154
Départ volontaire d'un agent simple d'un groupe . 157
Éjection d'un agent d'un groupe 158
Fusion de groupes d'agents 158
5.4 Outil de visualisation 161
5.5 Remarques 163
5.6 Morphologie et analyse du système 163
5.6.1 Proposition de morphologie : graphes 164
5.6.2 Analyse de graphes 164
Méthodes existantes d'analyse de graphes 164
Remarques 165
5.6.3 Types de graphes, hypothèses 165
5.6.4 Construction des graphes 166
5.6.5 Comparaison entre graphes 167
5.6.6 Module d'analyse 169
5.6.7 Retour sur le système 170

6 Application 171
6.1 Description de l'application 171
6.1.1 Les objets de l'environnement 172
6.2 Agentification 174
6.2.1 Rôle des agents 174
6.2.2 Connaissances réifiées dans les agents 176
Règles de composition 176
Métrique 176
6.3 Quelques résultats 178
6.3.1 Expériences 182
Sans contrôle 182
Avec contrôle 189
6.3.2 Comparaison de formes 194
6.4 Remarques et extensions 197

III Perspectives 199

7 Émotions artificielles 201
7.1 Émotions en biologie 201

7.1.1 Généralités 201
Types d'émotions 202
7.1.2 Théories sur les émotions 203
Théorie de James-Lange 203
Théorie de Cannon-Bard 204
Théorie de Schachter-Singer 204
Théorie de Lazarus 205
7.1.3 Mécanismes des émotions 205
7.1.4 Caractérisation des émotions 207
7.1.5 Quelques remarques 209
7.2 Émotions artificielles 211
7.2.1 Motivation 211
7.2.2 Quelques modèles 212
Travaux de Ventura 212
Modèle de Velásquez 215
Modèle de Gadanho 218
Travaux de A. Sloman 220
Travaux de M. Minsky 222
Quelques remarques 223
7.3 Pistes proposées 223
7.3.1 Motivation 223
7.3.2 Propositions 225
7.3.3 Oscillateur calculable 225
Interprétation d'un oscillateur 227
Illustration 229
Première expérience 229
Seconde expérience 232
Remarques 237
Plans et comportements basés sur l'émotion 237

Conclusion **241**

Liste des publications **245**

Annexes **245**

Table des figures **251**

Bibliographie **262**

Application 3 **263**
7.4 Introduction 263
7.5 Description Générale 263
7.6 Oz – Mozart 264
7.7 Description des robots utilisés 265
7.8 Interface robot–système 266
7.8.1 Connexions 266
7.8.2 Données 267
Capteurs 267
Effecteurs 267
7.9 Système de conduite comportementale 268
7.9.1 Agents 269
7.9.2 Agents d'interface 270
7.9.3 Agents aspectuels 270
7.9.4 Agents morphologiques 271
7.9.5 Agents d'analyse 272
7.9.6 Générateur de rôle 272
7.10 Divers 273
7.11 Conclusion 277

Résumé

La construction de systèmes multi-agents comportant de nombreuses entités, considérés comme des systèmes complexes, nécessite une méthode rigoureuse. Les méthodes orientées génie logiciel supposent qu'une analyse fine a priori est suffisante, sans vraiment tenir compte de l'aspect dynamique de tels systèmes. Les approches formelles, quant à elles, se confrontent à l'explosion combinatoire lors de l'exploration des états à vérifier. Une autre voie qui semble plus appropriée passe par l'observation a posteriori. Nous proposons une méthode de contrôle fondée sur la notion de morphologie : on analyse le comportement du système d'un point de vue géométrique. Un modèle du système complexe est construit a posteriori à partir de cette représentation. Nous proposons de piloter ce type de systèmes par tendance, contrôle non optimal, compromis intéressant. Par ailleurs, le paradigme systèmes multi-agents contrôlés par morphologie offre un cadre intéressant à l'étude des émotions artificielles.

Mots clefs

Systèmes multi-agents, systèmes complexes, morphologie, observation, contrôle, émotions artificielles.

Abstract

The engineering of large multi-agent systems, considered as complex systems, requires a method. Software engineering approaches suggest that a fine analysis of the problem is sufficient, however they do not consider the dynamical aspects of such systems. Formal approaches confront with the combinatory complexity of the states to verify. Another approach is based on observing the system a posteriori. We suggest a method of control based on morpholgy : we analyse the system from a geometric point of view. A model of the complex system is generated a posteriori from this representation. We propose to drive the system by tendency, an non-optimal control, an interesting compromise. In addition, the morphological multi-agent system offers an interesting framework for artificial emotions.

Keywords

Multi-agent systems, complex systems, morphology, observation, control, artificial emotions.

Introduction

Introduction

Présentation

Cette thèse se situe au carrefour du paradigme des systèmes multi-agents [Fer95], de la théorie morphologie [Tho72], de la théorie des systèmes dynamiques [AS92] et de la théorie du contrôle [Lew92]. Les travaux qui y sont développés s'inscrivent dans un projet à long terme, qui porte sur la construction d'un système de contrôle comportemental propre pour un robot, en exploitant la notion d'émotions artificielles fondée sur le contrôle morphologique de systèmes multi-agents.

Le paradigme multi-agent offre un outil de modélisation puissant. Ceci est en grande partie dû au fait que cette approche emploie des notions très intuitives et anthropomorphiques. Mais ce domaine souffre d'un manque de consensus, et peut-être même de formalisme, à propos de certaines définitions ou concepts, comme c'est le cas, par exemple, pour le terme « agent » qui présente une très vaste variété de définitions [FG96]. Le paradigme multi-agent offre quand même un vaste cadre d'investigation qui touche à beaucoup de domaines différents : informatique, réseau, démographie, physique, éthologie... Cet engouement pour les systèmes multi-agents a conduit à une myriade d'outils et de plate-formes, chacun proposant des fonctionnalités spécifiques.

Les systèmes multi-agents comportant un nombre important d'agents sont des systèmes complexes en ce sens que le comportement général du système est déterminé principalement par ses actions communicationnelles engagées de manière autonome par les agents. Le comportement global du système ne peut être déterminé par la seule considération de chacune de ses entités. Les communications, les mouvements et les interactions doivent nécessairement être pris en compte pour

le définir. Réaliser de tels systèmes nécessite une méthode rigoureuse. Il existe plusieurs méthodes de modélisation et de construction de systèmes multi-agents, par exemple : [WJK00, Del01, PW02, CCG+01, GMP02, Dem01]. Cependant, trop peu d'entre elles traitent explicitement le problème d'adéquation du comportement du système avec ses objectifs et encore moins son analyse et son contrôle en ligne. Nous pouvons distinguer trois grandes familles de méthodes : les méthodes orientées génie logiciel, les méthodes formelles et les méthodes fondées sur l'observation ou l'expérimentation.

Les méthodes orientées génie logiciel, qui forment une grande partie des approches proposées, parient sur le fait que l'analyse du problème a priori et la conception fine du système suffisent à réaliser un système multi-agent opérant avec un comportement adéquat pour ses objectifs. Mais un système complexe peut exhiber des comportements à l'exécution qui n'auraient pas été soupçonnés par ses concepteurs durant la phase de conception. De plus ces méthodes semblent omettre les aspects dynamiques et émergents de ce type de systèmes qui justement sont les caractéristiques intéressantes par rapport à des systèmes informatiques plus ordinaires.

Les méthodes formelles quant à elles sont plutôt soucieuses de valider le comportement d'un système multi-agent en spécifiant les propriétés du système sous une forme qui se prête aisément à la vérification automatique [RL04]. Cependant, le problème commun de toutes ces méthodes, qui se différencient essentiellement par la technique mise en œuvre dans la vérification automatique, est qu'elles se confrontent à une explosion combinatoire des calculs à effectuer, les rendant inefficaces pour valider de réels systèmes multi-agents de grande taille.

La troisième approche se fonde sur l'expérience et l'observation de tels systèmes, et sur la construction d'un modèle de ces systèmes à posteriori. Notre proposition se rattache à cette troisième approche. Le principal argument à l'encontre de cette approche est que l'observation du système ne prend en compte que quelques trajectoires du système et pourrait passer à côté de certains comportements [Ter01]. Néanmoins ce type d'approche reste applicable pour des systèmes comportant un grand nombre d'agents. Par ailleurs, les méthodes formelles ne se soucient pas vraiment de la question du contrôle (au sens pilotage), ni les approches orientées génie logiciel, puisqu'elles estiment le problème réglé d'avance.

L'originalité de notre approche réside dans la méthode employée pour l'analyse, dans le sens « observation », et pour le contrôle des systèmes multi-agents. Nous proposons d'analyser ces systèmes par une méthode géométrique, par l'ana-

lyse de la forme de la structure de l'organisation d'agents : sa morphologie. La notion de morphologie est relativement répandue dans des branches très différentes des Sciences : notamment en Sciences Physiques (espace des phases, forme des structures...) [Sch99] , en Mathématiques (stabilité structurelle, topologie...) [Tho72], en Biologie (forme des êtres vivants, des organes, de leur croissance, morphométrie...) [Boo91], en Statistiques (analyse en composante principale, moments...) [MH87]... En informatique, nous trouvons quelques applications, mais fort peu, par exemple [Ter01, Zal02], et très peu de travaux ont été menés sur ce point dans le cadre des systèmes multi-agents [Les00] ; ces systèmes s'apparentent pourtant très fortement aux systèmes complexes étudiés en Sciences Physiques (systèmes « non-linéaires », particulaires, chaotiques), et, en Sciences Physiques, la principale méthode d'étude de ces systèmes est justement la morphologie.

Le contrôle des systèmes complexes n'est pas le même que celui des systèmes qualifiés de simplement décomposables, de part leur nature même. Par exemple, l'observation des systèmes de régulation des organismes vivants, qui sont considérés comme des systèmes complexes, révèle qu'ils sont loin d'être des systèmes optimaux, mais que néanmoins ils réalisent assez bien leurs fonctions. Dans le domaine des systèmes dynamiques et chaotiques, les méthodes de contrôle développées pas les physiciens adoptent également cette approche [Sch99]. Bien que le comportement de ces systèmes soit imprévisible à long terme, et qu'ils soient difficiles à contrôler, ils ont développé des méthodes de contrôle qui font tendre un système pourtant chaotique dans un certain état stable, empruntant une certaine trajectoire. Mais il arrive que ces systèmes « décrochent » momentanément et, éventuellement, le contrôle reprend le dessus. Le contrôle ne se fait pas de manière stricte et absolue, mais il se fait « par tendance ». Par ailleurs vouloir faire du contrôle absolu ou réaliser un système multi-agent dont le comportement est prévu à l'avance n'est-il pas en contradiction avec la notion même d'agent ? Pourquoi leur prêterions-nous alors les caractéristiques d'autonomie et de pro-activité ?

Les travaux de la présente thèse s'inscrivent dans la lignée des travaux d'Alain Cardon [Car99, Car03], qui par ailleurs a dirigé la thèse de Stéphane Durand [Dur99] et de Franck Lesage [Les00], dans lesquels la notion de morphologie est exploitée pour piloter un système d'information à base d'agents. Les travaux de la présente thèse s'engagent dans la poursuite de cette voie, tout en essayant de la généraliser et de l'approfondir, en faisant le lien avec les autres branches de la Science, et en étudiant les autres méthodes de constructions de systèmes multi-agents.

Thèse défendue

Dans cette thèse nous défendons les points suivants :

- les systèmes multi-agents sont des systèmes complexes qu'il n'est pas évident de construire en utilisant seulement des méthodes de type génie logiciel classiques ; ces méthodes prennent difficilement en compte les aspects dynamiques et émergents de ces systèmes. Par ailleurs, les approches formelles d'analyse des systèmes multi-agents sont inadéquates pour des systèmes importants en terme de nombre d'agents et de complexité ; ces méthodes se confrontent à l'explosion combinatoire de l'espace à explorer ;
- si l'on s'autorise un contrôle non-optimal, les systèmes multi-agents peuvent être pilotés par tendance. C'est un compromis intéressant. Dans le domaine du vivant, on peut observer de réels systèmes complexes, et ces systèmes sont rarement des systèmes optimaux ;
- nous proposons de piloter, de contrôler, de tels systèmes à l'aide d'une représentation globale de l'état des organisations d'agents par l'observation et l'analyse morphologique. La morphologie consiste en une représentation géométrique du système. Cette appréciation du système semble d'autant plus pertinente que les systèmes multi-agents s'apparentent très fortement aux systèmes chaotiques étudiés en Physique, et que la méthode privilégiée d'étude des ces systèmes est la morphologie ;
- le contrôle morphologique des systèmes multi-agents offre un cadre pour l'investigation et la réalisation de la notion d'émotions artificielles.

Plan du manuscrit

Le manuscrit est composé de trois parties :

1. L'état de l'art :

   Chapitre 1 Le présent manuscrit commence par la présentation rapide des notions du paradigme multi-agent. Nous présenterons également quelques méthodes de construction et d'analyse de tels systèmes (orientées génie logiciel, méthodes formelles...). Par ailleurs, nous soulèverons le problème du contrôle de tels systèmes.

   Chapitre 2 Nous introduirons la notion de morphologie de façon générale, en Mathématiques, en Sciences Physiques, en Biologie et également en Informatique, puis plus particulièrement dans le domaine des systèmes multi-agents.

2. Les propositions :

   Chapitre 3 Dans ce chapitre nous verrons comment il est possible de représenter le comportement de systèmes multi-agents auto-adaptatifs d'un point de vue morphologique. Nous présenterons ensuite notre modèle d'analyse et de contrôle de systèmes multi-agents fondé sur la notion de morphologie. Le modèle sera présenté de façon assez générale.

   Chapitre 4 Dans ce chapitre nous présenterons la réalisation d'un contrôle morphologique appliqué à un système multi-agent réactif. Le système et la méthode effective de contrôle sont décrits et analysés. Des résultats expérimentaux y sont exposés.

   Chapitre5 Nous présenterons ensuite une méthode d'agentification particulière (« agentification aspectuelle ») et un système relativement générique exploitant cette agentification. Ce système a été effectivement implanté sous forme d'un prototype (en langage O'Caml).

   Chapitre 6 Ce chapitre présente l'étude d'un système de contrôle, exploitant le contrôle morphologique, la méthode d'agentification proposée et le système décrit dans le chapitre précédent, appliqué à un environnement simulé, où le système (piloté par un système multi-agents) doit agir sur l'environnement. Quelques premiers résultats d'expérimentations y sont présentés.

3. Les perspectives :

   Chapitre 7 Après avoir présenté quelques notions sur les émotions selon les biologistes et des travaux existants sur les émotions artificielles,

nous verrons que le contrôle morphologique des systèmes multi-agents offre un cadre d'étude intéressant pour les émotions artificielles, et comment notre approche se distingue d'une grande partie des autres modèles d'émotions artificielles.

Première partie

État de l’art

Chapitre 1

Systèmes multi-agents

Dans ce chapitre nous présentons rapidement quelques notions du paradigme multi-agent (section 1.1), notamment sur les agents et les systèmes multi-agents. Une brève méthodologie sera exposée (section 1.2) pour donner une vue générale sur le paysage de la recherche à propos des système multi-agents et dans le domaine de la construction de tels systèmes. Nous présenterons quelques méthodes de construction de tels systèmes (orientées génie logiciel, méthodes formelles...). Par ailleurs, nous soulèverons le problème de l'analyse (observation) et du contrôle de tels systèmes (section 1.3).

1.1 Notions d'agent et de système multi-agent

1.1.1 Agent

Le terme « agent » est un terme qui reste relativement vague, il n'y a pas de consensus sur la définition. Néanmoins, une définition communément admise en France, issue de J. Ferber dans [Fer95], semble préciser le terme :

> On appelle agent, une entité réelle ou abstraite qui est capable d'agir sur elle-même et sur son environnement, qui dispose d'une représentation partielle de cet environnement, qui, dans un univers multi-

> agent, peut communiquer avec d'autres agents et dont le comportement est la conséquence de ses observation, de sa connaissance et des interactions avec les autres agents.

Une autre définition, proposée par N. Jennings et M. Wooldridge dans [JW98] :

> Un agent est un système informatique, situé dans son environnement, et qui agit de façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu.

On confère les propriétés suivantes à tout agent :

- autonomie : capacité à agir sans aucune sollicitation externe ;
- réactivité : capacité à percevoir des changements dans son environnement et d'y réagir ;
- pro-activité : capacité à prendre des initiatives sans intervention extérieure ;
- sociabilité : capacité d'interaction avec d'autres agents (éventuellement humains) si nécessaire.

Nous allons considérer davantage l'agent comme un concept ou un outil de modélisation pour certaines classes de problèmes. Un agent ne représentera pas forcément une entité réelle, mais pourra représenter un aspect évolutif ou variable du problème étudié. Selon Alain Cardon dans [Car05] :

> On appellera agent une entité de conception proactive et évolutive permettant de réifier symboliquement, en tenant un rôle selon un point de vue strictement local, un certain aspect organisationnel du comportement, interne ou externe, d'un phénomène complexe.

Cette idée sera reprise ultérieurement lors de la présentation de la méthode d'agentification proposée dans le chapitre 3 et dans les exemples d'applications.

1.1.2 Système multi-agent

Un système multi-agent consiste en une ou des organisations d'agents, c'est à dire un ensemble structurellement cohérent où les agents communiquent entre

eux. L'agent est l'entité de base du système. L'organisation peut-être ouverte ou non, variable en taille, rigide ou modulable en structure. Un système multi-agent peut se définir comme un ensemble d'agents en interaction, capable de s'organiser d'une manière dynamique et adaptative [BD01]. Les systèmes multi-agents, comportant un grand nombre d'agents, sont des systèmes complexes en ce sens que le comportement général du système est déterminé principalement par ses actions communicationnelles engagées de manière autonome par les agents, celles-ci étant conjoncturelles et non strictement prévues lors de la conception. Le comportement global du système ne peut donc être simplement déterminé seulement par la considération de ses entités. Les communications, les mouvements et les interactions doivent nécessairement être pris en compte pour le définir entièrement.

Pour notre part, nous considérerons l'agent uniquement dans le contexte d'un système multi-agent. Nous ne considérerons pas le cas des agents isolés.

1.2 Méthodologie

1.2.1 Approches par conception

Nous dressons ci-dessous une liste non exhaustive d'outils de développement de systèmes multi-agents. Ce sont essentiellement des plate-formes multi-agents, ou des librairies de programmation qui offrent des fonctionnalités visant à faciliter la réalisation des systèmes multi-agents. Parfois ces outils sont accompagnés de méthodes de conception ou de développement :

- ADK [Try04] ;
- AgentSheets [Age] ;
- AgentTool [Kana] ;
- Bee-gent [Tos] ;
- CABLE [Loga] ;
- Comet Way [Com] ;
- CORMAS [CIR] ;
- Cougaar [DAR] ;
- DECAF [Del] ;
- DIMA [Gue96] ;
- EXCALIBUR Agent [AI] ;

- FIPA-OS [Emo] ;
- Gaia [WJK00] ;
- Grasshopper [IKV] ;
- IMPACT [Mar] ;
- JACK [Ltdb] ;
- JADE / LEAP [Mot] ;
- JAFMAS / JIVE [Cin] ;
- JATLiteBean [jat] ;
- JESS [Lab] ;
- Kaariboga [Kaa] ;
- Mace / Mace3J [Mac, GK02, GBH87] ;
- MadKit [Mada] ;
- MAML [Ltda] ;
- MaSE [Kanb, Del01] ;
- MASSYVE KIT [San] ;
- MAST [Madb] ;
- MESSAGE [CCG+01] ;
- NARVAL [Logb] ;
- Prometheus [PW02] ;
- RePast [Chi] ;
- RETSINA [Car] ;
- SimAgent [Bir] ;
- StarLogo [MIT] ;
- TuCSon [Bol] ;
- Tropos [GMP02] ;
- Voyager [Rec] ;
- Voyelles [Dem01] ;
- Xraptor [Gut] ;
- ZEUS [BTe] ;
- ...

Cette longue liste non exhaustive révèle l'intérêt porté au paradigme agent et aux systèmes multi-agents à l'heure actuelle. Ces outils et plate-formes sont pour une grande part orientés génie logiciel. En général, ces outils proposent :

- une aide à la conception (souvent proche de UML) ;
- des librairies de programmation ;
- parfois des outils de simulation ;
- un environnement de développement.

Dans la suite nous présenterons quelques uns de ces outils (méthodes ou plateformes).

Gaia

M. Wooldridge, N. Jennings et D. Kinny [WJK00] proposent la méthode Gaia pour l'analyse et la conception de systèmes multi-agents. Cette méthode générale prend en compte aussi bien le niveau macro (organisation) et le niveau micro (agent) des systèmes multi-agents. Le gros soucis de cette méthode est néanmoins que le système multi-agent résultant est relativement figé au cours de l'exécution (relation inter-agent et capacité de l'agent). La méthode Gaia se décline de la façon suivante (voir la figure 1.2.1) :

1. la phase d'analyse : qui consiste à établir des modèles préliminaires visant à comprendre ce que le système doit faire. On y définit :
   - la structure générale du système, à savoir s'il sera composé d'une ou de plusieurs organisations et, le cas échéant, comment elles sont couplées,
   - le modèle de l'environnement, qui représente l'ensemble des ressources mises à disposition des agents, des objets qu'ils peuvent manipuler, modifier, créer,
   - le modèle préliminaire des rôles, qui comprend l'ensemble des fonctionnalités ou compétences de base du système réifiées dans les agents nécessaires afin que le système puisse atteindre son but,
   - le modèle préliminaire d'interactions, qui représente l'ensemble des dépendances et les relations entre les rôles,
   - les règles organisationnelles. Ces règles capturent les relations générales entre les rôles, et entre les protocoles au niveau de l'organisation ;

2. la phase de conception : cette phase comprend les choix à faire sur les caractéristiques du système pour effectivement réaliser ce qui a été saisi lors de la première phase, c'est à dire :
   - la conception de l'architecture, où est fait le choix de la structure organisationnelle. On y opère le repérage et l'application des patrons organisationnels autant que possible. L'organisation est représentée à l'aide d'une notation formelle et sous forme graphique, et on complète les modèles de rôles et les modèles d'interactions,
   - la phase de conception fine, qui est la phase responsable de l'identification précise des rôles des agents, ainsi que du modèle des services que

chaque agent offre aux autres agents.

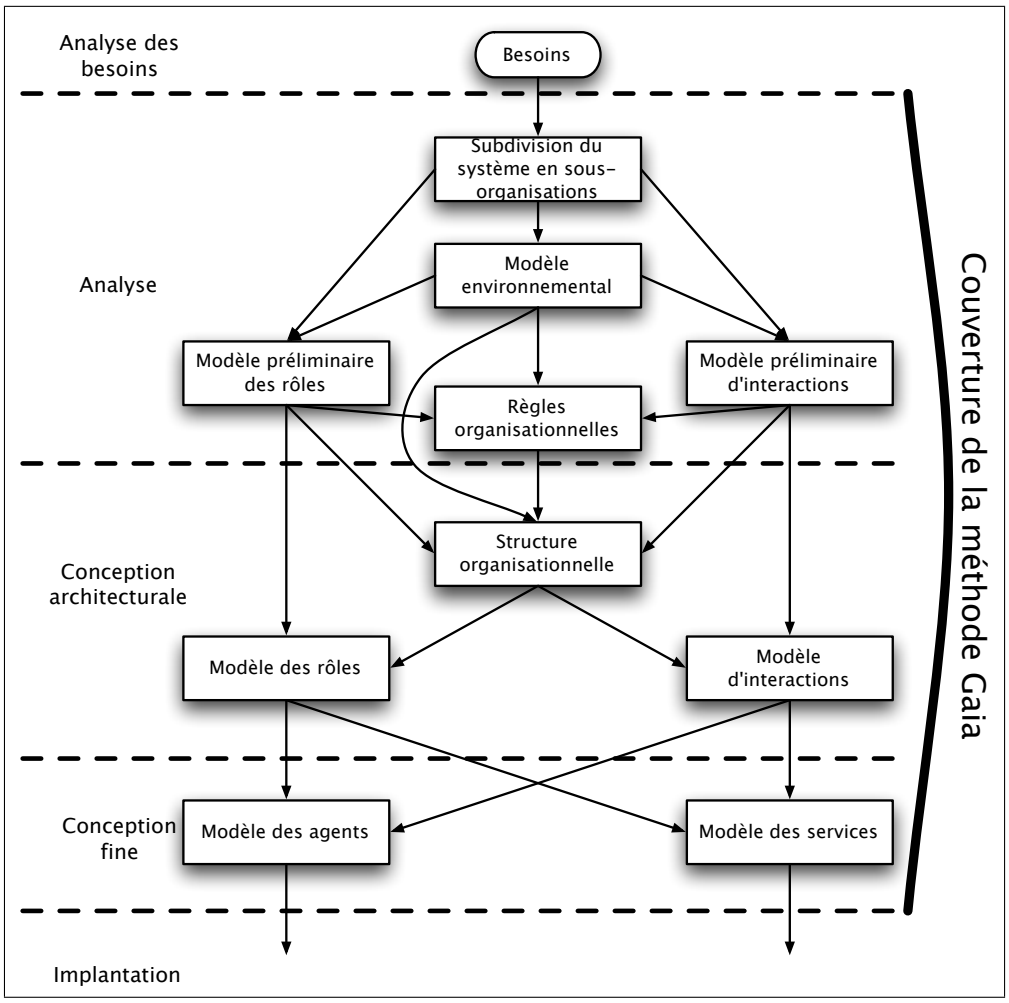


Figure 1.2.1 – *La méthode Gaia couvre les phases d'analyse, de conception architecturale et de conception fine.*

Le rôle d'un agent, qui représente la fonction de l'agent au sein de l'organisation, est défini par quatre attributs :

- les responsabilités : ce que le rôle apporte comme contribution au système ;
- les permissions : ce que le rôle est autorisé à faire, en particulier l'information à laquelle il a accès ;

- les activités : les tâches réalisées par le rôle sans la moindre interaction avec les autres rôles ;
- les protocoles : un motif particulier d'interaction.

Beaucoup de choses semblent à priori relativement figées lors de la conception dans cette méthode, notamment les relations entre agents. Néanmoins, les auteurs affirment qu'à l'issue de la phase de conception, la méthode fournit aux développeurs un ensemble de classes d'agents bien définies, à implanter et instancier selon les modèles d'agents et de services pour réaliser le système.

MaSE et agentTool

M. Wood et S. Deloach [Del01] proposent la méthode MaSE (« Multiagent Systems Engineering »). Avec l'outil MaSE, le développeur est accompagné depuis la conception du système jusqu'à son implantation. La démarche est segmentée en sept étapes linéaires, réunies en deux phases (voir la figure 1.2.2) :

1. en phase d'analyse, qui comprend :
   - l'étude des besoins, consistant en la transformation des spécifications du système en une hiérarchie de buts,
   - l'étude de cas, afin de mieux préciser les besoins et les interactions entre rôles à l'aide de diagrammes de séquence,
   - le raffinement des rôles : élaboration des rôles qui seront responsables et nécessaires à l'atteinte des buts. Typiquement, un rôle sera affecté à un but, mais il se peut qu'un ensemble de but soit rattaché à un seul rôle. Par suite, les tâches sont définies : les tâches définissent comment les buts sont atteints par le biais des rôles précédemment définis. Les tâches sont présentées sous forme de diagramme d'états ;
2. en phase de conception, qui se décline de la façon suivante :
   - la création des classes d'agents, par association des rôles aux classes d'agents sous forme de « diagrammes d'agents ». Ces diagrammes sont similaires aux diagrammes de classes d'objets mais la sémantique des relations entre entités est différente, elle est de niveau conceptuel plus élevé,
   - la construction des échanges, phase qui définit le protocole de coordination sous forme de diagramme d'états qui spécifie les interactions entre les agents ;
   - l'assemblage des classes d'agents : c'est la phase qui précise les fonctionnalités interne des agents,

– et finalement, la conception du système.

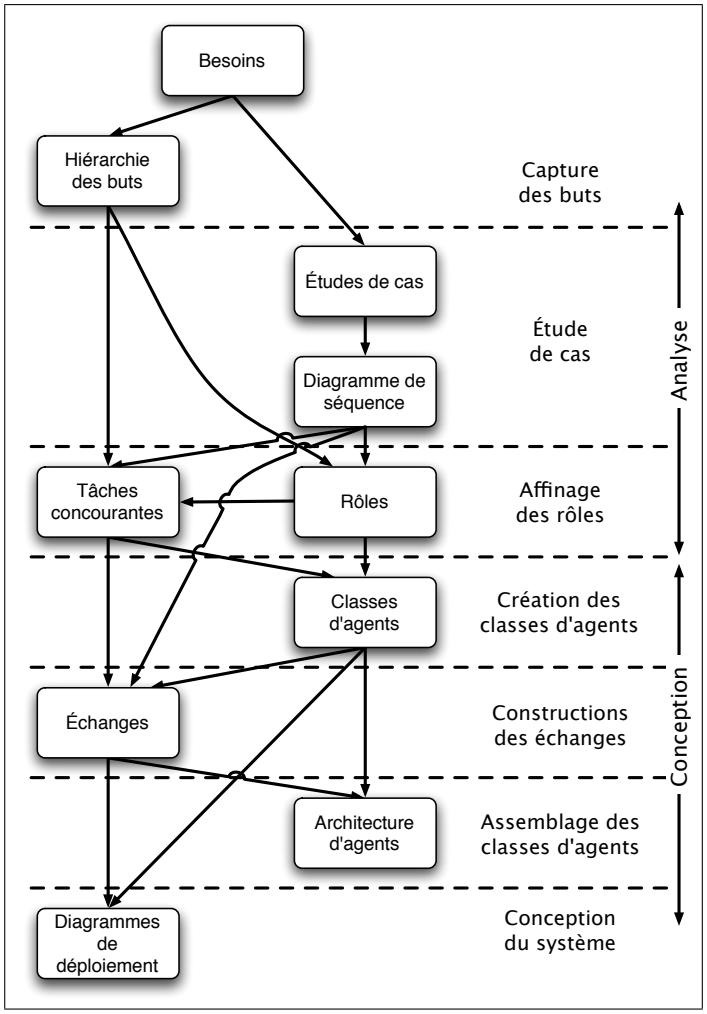


Figure 1.2.2 – *La Méthode MaSE couvre les phases d'analyse et de conception.*

Cette méthode est indépendante de l'architecture de l'agent, du langage de programmation ou du modèle de communication. Néanmoins, cette méthode s'arrête à la conception du système. Cette méthode ne propose pas de méthode de contrôle.

Le système agentTool est une plate-forme qui tente d'instancier la méthode MaSE. AgentTool accompagne le concepteur pour les sept étapes de MaSE et propose aussi un mécanisme d'automatisation pour transformer les modèles d'analyse en modèles de conception.

MESSAGE/UML

La méthode MESSAGE/UML [CCG+01, EUR01b, EUR01a] (« Methodology for Engineering Systems of Software Agents ») est en fait une sorte d'extension de UML à laquelle de nouveaux concepts et une notation spécifique sont rajoutés afin de pouvoir mieux spécifier des concepts agent. Les diagrammes étendent les classes UML et les diagrammes d'activités.

Les concepts orientés agent de MESSAGE (voir figure 1.2.3) ajoutés à UML sont :

- agent : entité atomique autonome capable de réaliser une fonction ;
- organisation : groupe d'agents opérant ensemble pour un but commun ;
- rôle : la distinction entre rôle et agent est similaire à celle entre une interface et une classe d'objet, le rôle décrivant les caractéristiques externes de l'agent dans un contexte donné ;
- ressource : représente toute entité non autonome ;
- tâche : activité définie avec une pré-condition et une post-condition. Si les pré-conditions sont satisfaites et que la tâche est activée, on peut s'attendre à ce que les post-conditions deviennent vraies. Cette notion permet de créer des chaînes de causalités ;
- interaction et protocole : spécifie comment les agents interagissent lorsqu'ils tentent de coopérer pour atteindre un but commun ;
- but : associe un agent avec une situation ;
- entité informationnelle : c'est une donnée, une information (entité passive) ;
- message : un message, non pas au sens UML (appel de méthode d'un objet) mais davantage imagé : comme une information passée à un autre agent.

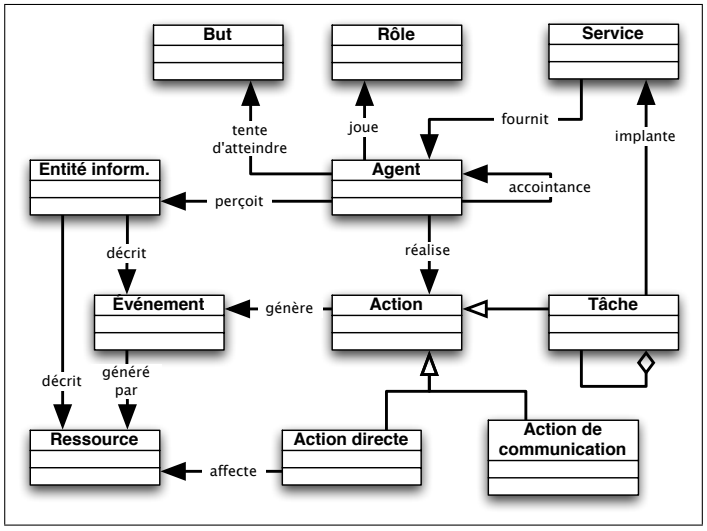


Figure 1.2.3 – *Les concepts de la méthode MESSAGE/UML.*

L'analyse est séparée en deux niveaux, que l'on nomme ;

– « niveau 0 » ;
– et « niveau 1 ».

Chacun de ces niveaux a pour but de construire des modèles spécifiques et de les raffiner (voir figure 1.2.4).

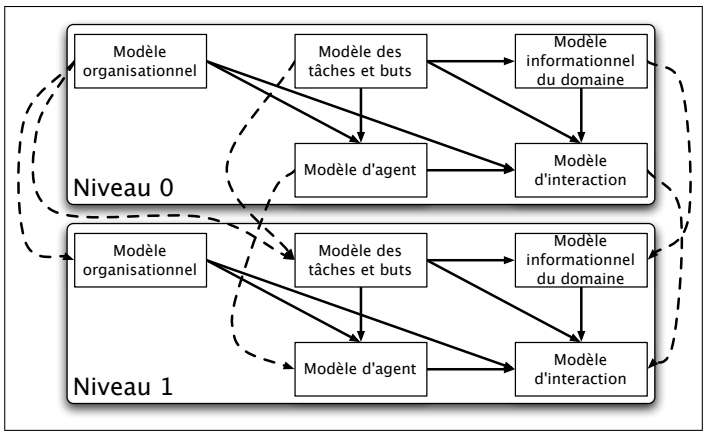


Figure 1.2.4 – *Les deux niveaux d'analyse de la méthode MESSAGE.*

Au niveau 0, le modèle définit l'ensemble des interactions possibles entre le système multi-agent et l'environnement externe dans lequel le système est plongé :

- le modèle d'organisation décrit la structure et le comportement d'un groupe d'agents ayant un but commun. À ce niveau 0, ce modèle procure une vue du contexte dans lequel le système opère en termes d'agents et de relations de pouvoir ;
- le modèle buts/tâches décrit comment les buts/tâches de haut niveau sont décomposés en sous-buts/sous-tâches de niveau inférieur. À ce stade, il contient un ensemble de tâches et de buts pour l'organisation dans laquelle le système est plongé ;
- le modèle d'agent consiste en une description des objectifs, des relations et des autres attributs des agents et des rôles. Le système multi-agent est vu comme s'il était un agent unique à ce niveau 0 ;
- le modèle du domaine encapsule les informations sur le domaine d'application ;
- le modèle d'interaction saisit les moyens d'échange d'informations entre les agents et leur environnement.

Au niveau 1 de l'analyse :

- le modèle d'organisation est précisé à ce niveau, notamment par l'analyse des autres modèles préliminaires ;
- le modèle buts/tâches est raffiné et les tâches sont détaillées, grâce à l'analyse du modèle d'organisation des niveaux 0 et 1 ;
- le modèle d'agent est une décomposition du modèle de niveau 0 en un ensemble d'agents capables de fournir le même ensemble de services qu'au niveau 0. Cet ensemble est dérivé de l'organisation de niveau 1 ;
- le modèle du domaine étend le modèle de niveau 0 par l'ajout d'objets et de ressources nécessaires au bon fonctionnement des nouveaux agents identifiés ;
- le modèle d'interaction permet de spécifier les échanges entre les nouveaux agents identifiés et de préciser les accès aux nouvelles ressources.

Tropos

Dans [GMP02, Tro], Fausto Giunchiglia, John Mylopoulos et Anna Perini présentent la méthode Tropos. Cette méthode couvre toute les phases du cycle de développement d'un logiciel. D'après leurs auteurs, elle est fondée sur deux idées clefs :

- la notion d'agent ou « acteur » (voir figure 1.2.5) est employée durant toutes les phases du développement du logiciel (de la conception à l'implantation) ;
- Tropos couvre les toutes premières phases de l'analyse, et donc, doit permettre une plus profonde compréhension de l'environnement dans lequel le logiciel doit fonctionner ainsi qu'une bonne compréhension du type d'interaction qu'il peut y avoir lieu de définir entre les agents.

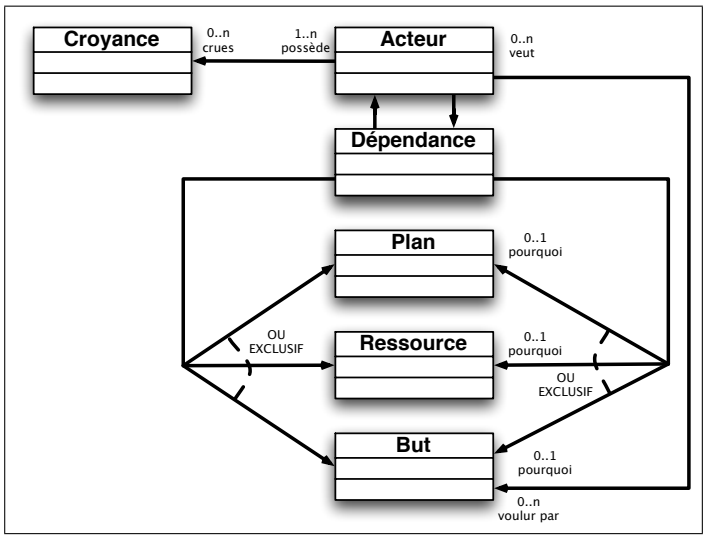


Figure 1.2.5 – *Diagramme UML spécifiant la notion d'acteur dans la méthode Tropos.*

Les notions introduites par la méthode Tropos sont :

- but : ce que les acteurs du système tentent de réaliser ;
- dépendance : les dépendances décrivent les relations entre acteurs lorsque ces derniers interagissent afin d'atteindre un but ou de mettre un plan à exécution ;

- plan : une façon précise d'atteindre un but particulier ;
- ressource : donnée ou ressource physique mise à disposition des acteurs, ou fournie par d'autres acteurs ;
- aptitude : capacité d'un acteur à définir, choisir et exécuter un plan afin d'atteindre un but ;
- croyance : connaissance d'un acteur.

Les étapes de déroulement de la méthode Tropos sont :

- l'analyse préliminaire : étude du problème visant à établir le modèle organisationnel comprenant les acteurs importants, ainsi que leurs buts respectifs ;
- l'analyse approfondie : le système à réaliser est décrit en prenant en compte son environnement opérationnel. Le système est considéré comme un acteur et les relations entre le système et les autres acteurs sont modélisées. Ces relations définissent les besoins du système ;
- la conception de l'architecture : l'architecture globale du système est décrite en terme de sous-systèmes et en spécifiant les relations entre ces entités. De plus les aptitudes de chaque acteur nécessaires pour parvenir à leur buts sont spécifiées ;
- la conception fine : spécification de chaque composant du système en terme d'entrées, sorties et contrôle. ;
- et finalement la phase d'implantation du système.

Prometheus/PDT

La méthode Prometheus est proposée par L. Padgham et M. Winikoff [PW02, RMIa, RMIb]. PDT est un outil supportant la méthode Prometheus. Il s'agit d'une application Java. La méthode Prometheus est segmentée en 3 étapes (voir figure 1.2.6) :

1. la spécification du système, où sont effectués :
   - l'identification des buts et sous-buts,
   - la réalisation de cas d'études,
   - la spécification de l'interface système de l'agent avec l'environnement par rapport aux actions et percepts,
   - l'identification des fonctionnalités,
   - ainsi que l'identification des types de données lues et écrites par ces fonctions ;

2. la conception de l'architecture, qui consiste à :
   - regrouper les fonctionnalités afin de déterminer les types d'agents en se basant sur le couplage des données,
   - et à faire la conception des protocoles d'interactions selon les cas d'études par le biais de diagramme d'interactions ;
3. finalement, la conception fine, qui comprend :
   - la conception des diagrammes de processus,
   - la définition de la structure interne des agents,
   - la définition des détails des vénements, des plans et des types de données.

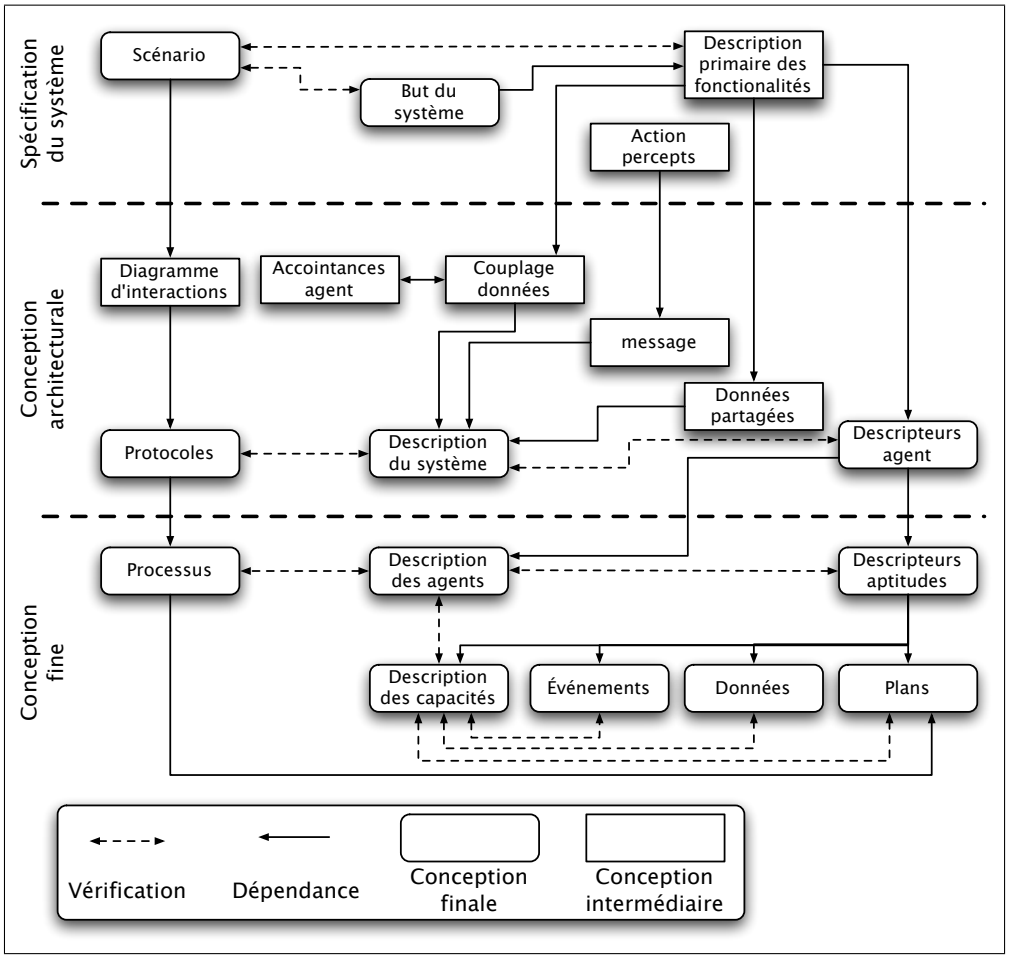


Figure 1.2.6 – *Méthode Prometheus*

Voyelles

L'approche Voyelles [Dem01] est une méthode d'agentification développée au sein de l'équipe Magma. La conception d'un système multi-agent passe par la décomposition d'un tel système en quatre « briques », selon les voyelles les « AEIO » :

- agent : entité de base du système ;
- environnement : l'espace dans lequel les agents sont plongés ;
- interactions : les interactions entre les agents, sur le plan infrastructure, langage, protocole... ;
- organisations : la structure des relation entre agents.

Par ailleurs, la méthode Voyelles repose sur 3 déclarations, qui sont :

1. un système multi-agents est composée de « Voyelles » (A,E,I,O) ;
2. la fonctionnalité d'un système multi-agent n'est pas seulement la somme des agents, mais la composante collective due aux interactions et la composante émergente du système en sont indissociables ;
3. un principe de récursion permet de concevoir des systèmes multi-agents comme des agents.

Les trois phases traditionnelles de construction de systèmes en génie logiciel, se retrouvent dans la méthode Voyelles (voir figure 1.2.7) :

- l'analyse porte sur le domaine d'application et le type du problème. Elle consiste en la décomposition du problème en voyelles, c'est-à-dire les différentes composantes du système. À un plus haut niveau d'abstraction, les entités identifiées devront répondre aux principes de récursion ou d'émergence. À la fin de l'analyse, le problème est donc « voyellisé » ;
- la conception, dans Voyelles, correspond à l'identification des modèles à utiliser pour chacune des voyelles. Ici encore, les principes de haut niveau (récursion et émergence) doivent être utilisés ;
- l'implantation, cette phase consiste en l'instanciation des modèles, en utilisant des plates-formes et des langages choisis. Le système implanté, est alors exécuté, évalué (test et validation) et repensé en cas d'inadéquation avec les besoins exprimés par le type de problème et le domaine d'application.

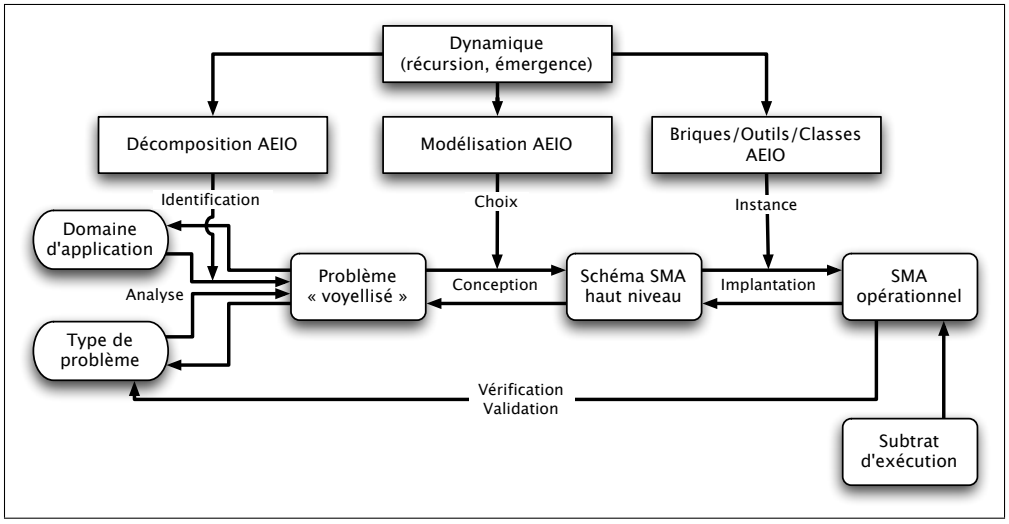


Figure 1.2.7 – *La méthode Voyelles.*

La thèse de Pierre-Michel Ricordel [Ric01], qui étudia l'application de la méthodes Voyelles, a donné lieu :

- à une spécification de la plate-forme « Volcano » de construction de systèmes multi-agents basée sur l'approche voyelles [RD02b, RD02a] ;
- au développement d'un langage de description de systèmes multi-agents, « Madel », permettant de décrire l'assemblage des briques formant un système multi-agent [RD01] ;
- au développement d'un outil d'assemblages de briques « Vésuve », outil de déploiement automatique « Lava » [Ric01] ;
- à des tests et validations de la plate-forme sur des applications telles que Dag (prise de rendez-vous), SMAM (Systèmes Multi-Agents Minimaux) et lors de la participation à la coupe Robocup [Ric01].

MADKit/Aalaadin

MadKit [Gut01, GF00, GF98a, GF98b, Mada] est une plate-forme multi-agents écrite en Java basée sur un modèle organisationnel. Son développement fut initié par Olivier Gutknecht et Jacques Ferber du LIRMM (Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier). MadKit est avant tout une plate-forme d'exécution de systèmes multi-agents, utilisant un micro-noyau agent. Le modèle organisationnel sous-jacent est le modèle Aalaadin [FG99]. Ici, l'organisation n'est pas le résultat émergent de l'activité du système, mais un moyen pour lui d'atteindre son but. Le modèle « AGR » (Agent–Groupe–Rôle) est défini comme dans la figure 1.2.8. Un agent fait partie d'un groupe et y joue un certain rôle qui représente sa fonction abstraite.

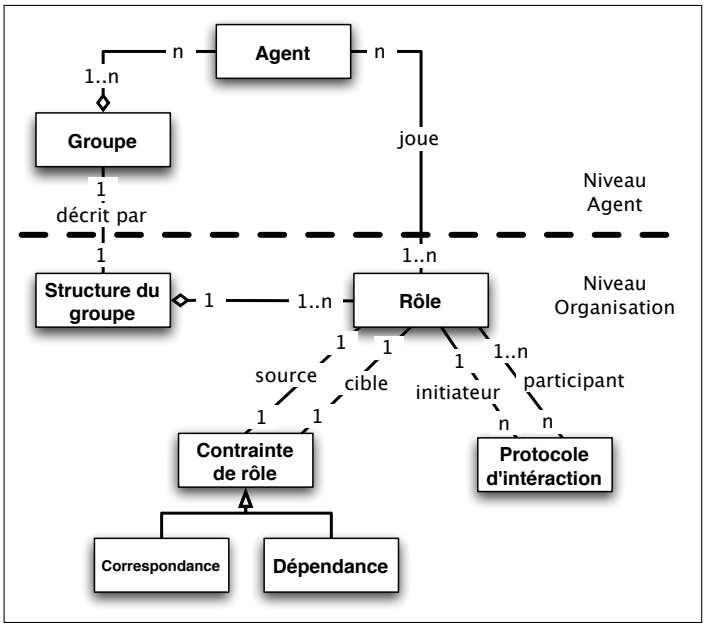


Figure 1.2.8 – *Méta-modèle UML de AGR (Agent–Groupe–Rôle).*

Les trois notions centrales sont :

– l'agent : le modèle n'impose pas de contrainte particulière sur l'architecture interne des agents. Un agent est uniquement spécifié comme une entité communicante qui joue un ou plusieurs rôles dans un ou plusieurs groupes. Cette définition très générale est volontaire : cela laisse la liberté aux concepteurs

de spécifier la définition qui leur semble la plus adéquate pour leur application ;
- le groupe : un agent fait partie d'un ou plusieurs groupes. Un groupe est un ensemble atomique d'agents. La notion de groupe permet d'étiqueter un ensemble d'agents. Tout agent a la possibilité de créer un groupe ;
- le rôle : le rôle est l'abstraction de la fonction de l'agent (ou bien le service qu'il fournit ou son identification) au sein d'un groupe. Un agent peut avoir plusieurs rôles, et chacun des rôles est local à un groupe. Un agent peut demander un rôle dans un groupe, mais celui-ci ne lui sera pas forcément attribué.

La méthode Aalaadin peut se résumer en trois phases :

1. l'analyse. Elle permet d'identifier les fonctions du système et les dépendances au sein de communautés identifiées. Il convient de définir quels sont les mécanismes de coordination et d'interaction entre les entités d'analyse ;
2. la conception. Cette phase consiste en l'identification des groupes et des rôles dans des diagrammes de structures organisationnelles. Les protocoles d'interactions alors identifiés entre rôles sont décrits dans des diagrammes organisationnels de séquences. L'identification des objets et des actions sur ces objets permet de commencer à décrire les attributs des agents ;
3. la réalisation. Cette phase comprend : le choix de l'architecture d'agent, la gestion des entités du domaine permet d'implanter le système.

Une caractéristique importante de MadKit est son micro-noyau agent (voir figure 1.2.9). L'agent micro-noyau se charge de la création des groupes et de leur gestion et de la distribution des messages locaux. Tous les services hors ceux assurés par le micro-noyau sont implantés par des agents à part entière. Ceci vient à la fois d'un souci d'unicité de modèle, mais également de mise à l'épreuve des systèmes multi-agents comme modèle de programmation [Gut01]. Ce centrage sur l'infrastructure simplifie les phases de développement et de déploiement. Le principe essentiel de MadKit est d'utiliser partout où cela est possible la plateforme pour son propre fonctionnement.

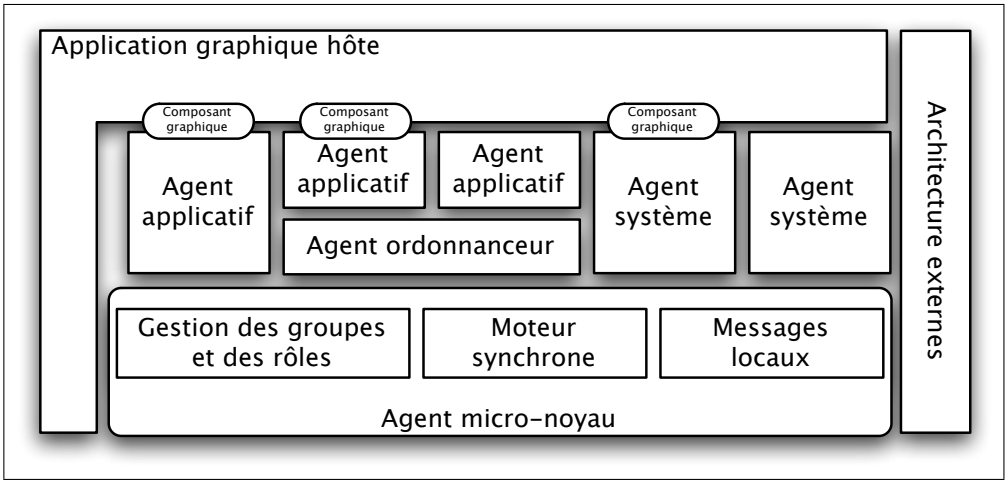


Figure 1.2.9 – *La structure générale de Madkit.*

Le déploiement dans MadKit se déroule dans la « G-Box », où les agents peuvent être créés, modifiés et retirés, par le biais d'une interface graphique (voir figure 1.2.10). Un mode sans interface graphique est également disponible pour simplifier le déploiement. La G-Box permet la configuration dynamique des propriétés modifiables des agents. Les agents sont encapsulés dans des archives Java et peuvent être mélangés avec toutes sortes d'autres agents, ce qui permet de réutiliser facilement des agents génériques. Plusieurs G-Box peuvent être interconnectées par réseau pour distribuer les traitements.

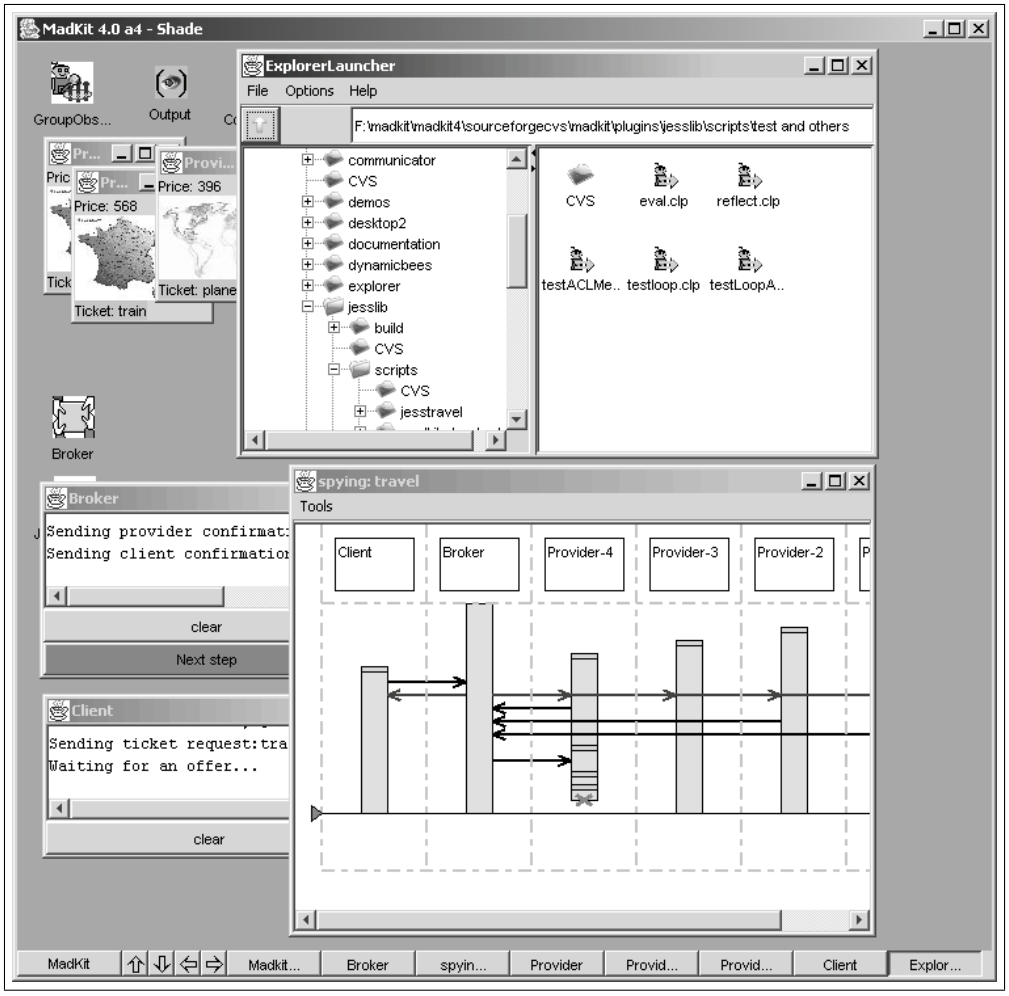


Figure 1.2.10 – *Capture d'écran de la G-Box de Madkit (version 4.0). On peut y voir deux types d'agents : l'agent « Spy agent » qui génère des diagrammes de séquences dynamiques ; et l'agent « MadKit Explorer » qui permet d'explorer l'arborescence des dossiers et de lancer de nouveaux agents.*

AMAS/Adelfe

De façon similaire à MESSAGE/UML, la méthode ADELFE [ade] (Atelier pour le DÉveloppement de Logiciels à Fonctionnalité Émergente), élaborée dans le cadre de la thèse de thèse de Gauthier Picard [Pic04], est une extension de UML qui tente de prendre en compte les notions associées au paradigme multi-agent. La méthode ADELFE a en fait été développée afin de fournir un outil applicatif de la théorie des AMAS [JP03] (« Adaptive Multi-Agent Systems »).

Selon la théorie AMAS, pour tout système fonctionnellement adéquat réalisant la fonction souhaitée, il existe au moins un système dont les agents sont en interactions coopératives. Cela signifie que pour concevoir un système réalisant la fonction attendue, il suffit que les agents le composant aient pour attitude sociale la coopération [JP03].

Le processus d'ADELFE est basé sur le Rational Unified Process et y ajoute des activités spécifiques à l'ingénierie orientée agent. Les notations sont une extension des notations UML et A-UML. La méthode ADELFE propose une plateforme comprenant un outil de modélisation graphique ; une bibliothèque de composants permettant des simulations et un prototypage rapide et une technique d'auto-assemblage des composants logiciels que sont les agents adaptatifs.

La méthode ADELFE comprend (voir figure 1.2.11) :

1. une phase d'analyse.
   - Dans cette phase on vérifie la pertinence d'une approche multi-agent. Cela entend l'étude de critères tels que :
     - complexité grande,
     - absence d'autres méthodes,
     - frontières mal définies,
     - distribution physique ou fonctionnelle,
     - environnement évolutif,
     - besoin d'interactions–coopération entre entités.
   - Puis, la suite de l'analyse est similaire à celle de UML ;
2. une phase de conception. Pour la conception d'un agent, ADELFE propose une description d'agent générique composée de sept modules :
   - communication avec les autres agents,
   - communication avec l'environnement,
   - croyances sur lui-même,

– croyances sur les autres agents,
– croyances sur son environnement,
– compétences,
– attitude sociale coopérative.

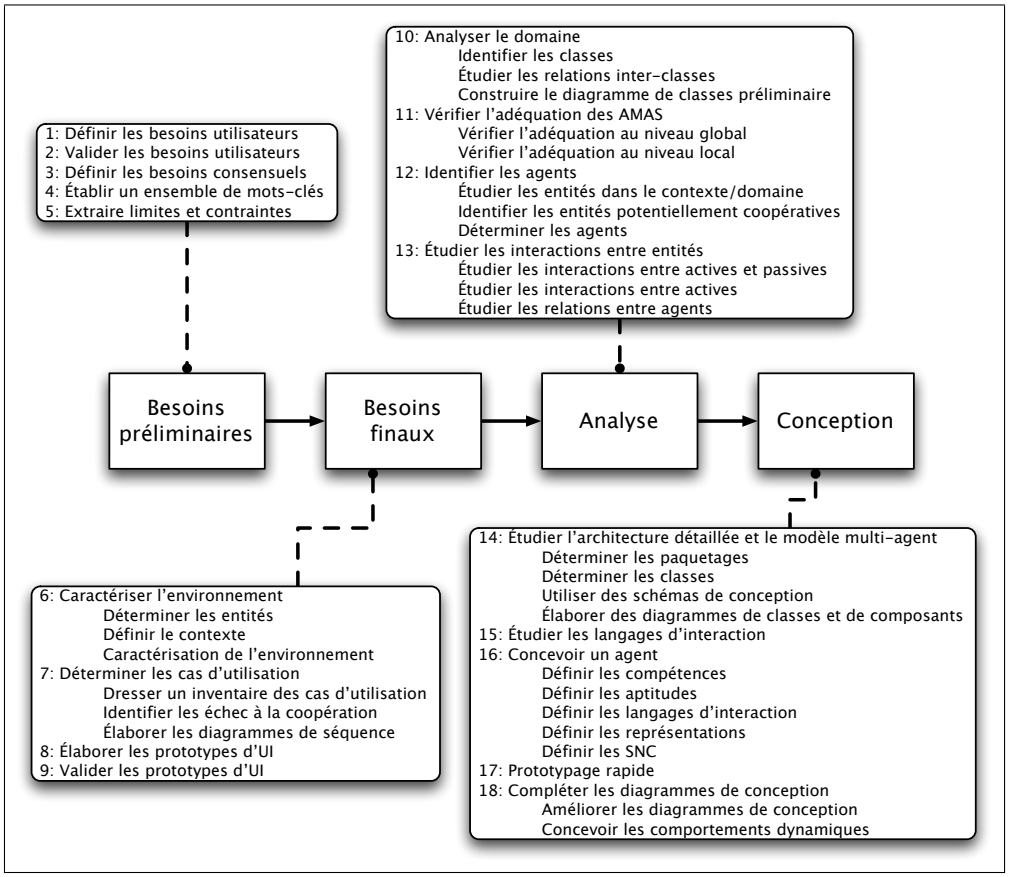


Figure 1.2.11 – *La méthode ADELFE.*

Voici quelques applications de la théorie AMAS et de la méthode ADELFE :
– ABROSE : « Agent-based Brokerage Services in Electronic Commerce », un système de courtage en ligne [GG00, GLPG00] ;
– ANTS : simulation de comportement d'une fourmilière lors des tâches de fourragements [TFG+99] ;
– STAFF : « Software Tool for Adaptive Flood Forecast », système de prévision de crues [JPMPPC03].

ZEUS

ZEUS [HSNL98, AT00, BTe] est un système multi-agent libre qui consiste en un environnement intégré pour la construction de systèmes à base d'agents collaboratifs. Ce système a été développé par l'Agent Research Program du British Telecom Intelligent System Research Laboratory.

La plate-forme ZEUS, totalement développée en Java. consiste en fait en une collection de classes Java. Ces classes sont catégorisées en 3 groupes fonctionnels :

- une librairie de composants,
- des outils de développement
- et des outils de visualisation.

ZEUS propose une grande panoplie de module pour construire des applications à base d'agents. Néanmoins étant une plate-forme relativement générique, la création d'applications spécifiques nécessite d'importantes augmentations. La documentation de ZEUS est abondante, et insiste particulièrement sur l'importance des aspects méthodologiques de ZEUS.

La panoplie d'outils proposée dans ZEUS inclut :

- un outil de rapport qui permet de représenter sous forme de graphes la décomposition des tâches et l'état d'exécution des différentes tâches ;
- un outil vidéo permettant de visualiser des simulations ;
- un outil d'analyse pour observer l'état interne des agents ;
- un outil qui permet de visualiser les messages échangés entre les agents ;
- un outil d'analyse statistique qui permet de recueillir des statistiques à l'échelle des organisations ;
- un outil de pilotage pour modifier l'état interne des agents.

La figure 1.2.13 illustre la structure d'un agent ZEUS.

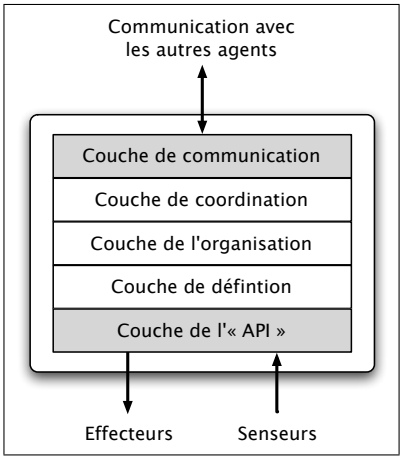


Figure 1.2.12 – *Diagramme d'un agent générique.*

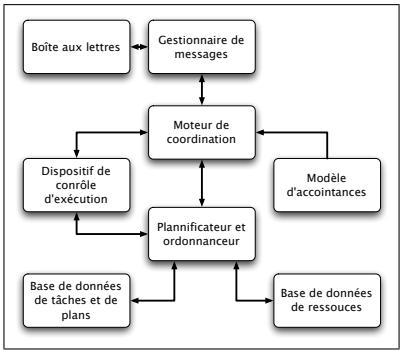


Figure 1.2.13 – *Architecture d'un agent de ZEUS.*

La méthode ZEUS utilise une décomposition en quatre étapes pour la construction d'agents que l'on retrouve dans beaucoup d'autres méthodes d'analyse :

1. analyse du domaine ;
2. conception ;
3. réalisation ;
4. support d'exécution.

Quelques applications (ou prototypes) développées avec ZEUS :

– un système de réservation de voyages, où un agent planifie des voyages pour les utilisateurs, négociant les réservations des hôtels, des billets d'avions, des taxis et des trains [NCN98] ;
– une application de commerce électronique où les agents négocient selon des stratégies différentes [Nwa98].

1.2.2 Approches par analyse, observation

Il existe un courant de recherche qui s'intéressent aux propriétés des systèmes multi-agents opérants. En particulier, les méthodes formelles et les méthodes fondées sur l'observation.

Méthodes d'analyse formelle

En quelques mots, les méthodes formelles d'analyse de systèmes multi-agents s'appuient sur un formalisme permettant de transcrire les propriétés de systèmes multi-agents et ensuite de travailler sur les propositions logiques ainsi construites pour déduire ou vérifier des propriétés du système. Citons quelques uns des formalismes les plus souvent employés, par exemple :

– la logique intentionnelle de Cohen et Levesque [CL90] ;
– le cadre des agents BDI (« Belief, Desir, Intention ») de Rao et Georgeff [RG91] ;
– le formalisme MATL (« MultiAgent Temporal Logic ») de Benerecetti, Giunchiglia et Serafini [BPST02] ;
– et LORA (« Logic of Rational Agents ») de Wooldrige [Woo00].

Les méthodes formelles de vérification se présentent sous deux approches :

– démonstration de théorèmes
– et vérification de modèles.

Les approches basées sur la démonstration de théorèmes consistent en la spécification d'un programme dans une logique formelle. Ainsi la vérification du fait que le programme en question opère d'une certaine manière revient à vérifier si

une formule particulière – celle représentant la propriété à vérifier – est un théorème de la logique considérée. Un des problèmes de ces méthodes qui opère par démonstration de théorème est que les logiques utilisées nécessitent d'importantes ressources computationnelles, ce qui limite leur champ d'application.

Les méthodes basées sur la vérification de modèles furent proposées comme alternative à celles basées sur la démonstration de théorème. Dans ce cadre, un système est décrit par un modèle temporel par le biais d'un programme représentant le système, qui est écrit dans un langage se prêtant aisément à la vérification de modèle, tel que SMV, par exemple [nus]. Ainsi, vérifier si un système satisfait une propriété, revient à vérifier formellement une formule particulière ; cependant cette méthode se heurte également à un problème de trop grand nombre d'états à vérifier. Pour parer à ce problème d'explosion combinatoire, des méthodes symboliques furent développées. Dans cette approche, le modèle temporel est représenté symboliquement par des structures logiques. Les travaux se distinguent ensuite essentiellement sur la technique employée pour faire les vérifications. Ces dernières années ont vu l'apparition de vérificateurs de modèles tentant de traiter de plus grands nombres d'états, par exemple, ceux mettant en œuvre des techniques utilisant des arbres de décisions binaires [Bry86, RL04].

Méthodes fondées sur l'observation

Il existe des méthodes d'analyse similaires aux méthodes formelles mais où la notion de morphologie est un peu présente. On y cite notamment la notion de morphisme et la notion d'enveloppe.

Les méthodes fondées sur l'observation consistent à faire observer un système multi-agent alors qu'il est en fonctionnement pour en tirer des conclusions sur son comportement. Teràn [Ter01] pose alors la question suivante : quelles expériences, quels scénarios, on doit mener ? En d'autres termes, si l'on considère le comportement global du système comme la composition des actions de chacun des agents, le choix du scénario revient à déterminer quelles actions les agents effectuent, et quels sont leurs paramétrages. Il y a globalement deux approches [Ter01] : l'utilisation de scénario et l'approche par la méthode de Monte Carlo. Le problème avec les scénarios c'est que l'on explore une trace à la fois, une trace étant définie comme la séquence de tous les choix opérés par tous les agents du système étudié. La méthode de Monte Carlo permet en revanche, de par sa nature, de faire des inférences statistiques et d'avoir un modèle probabiliste, mais

elle ne fournit pas de preuve. Teràn [TEW, OE02, Ter01] propose une méthode de recherche exhaustive des trajectoires, pour résoudre ce problème. Mais il précise que cela revient à peu près à faire de la preuve formelle. Néanmoins l'auteur utilise des notions intéressantes comme celle de morphisme et celle d'enveloppe de tendances.

En quelques mots, en s'appuyant sur le formalisme de Zeigler [Zei76], qu'il étend pour prendre en compte le fait que la structure d'un système multi-agents peut être dynamique, Teran génère l'arbre d'exploration des trajectoires (voir figure 1.2.14) qu'il fournit ensuite au système en entrée (la séquence des choix des agents). L'espace des trajectoires du système est construit par le produit cartésien de tous les espaces de choix des agents et par la dimension du temps. Il observe alors le résultat produit par le système, où il détermine ces tendances (voir figure 1.2.15).

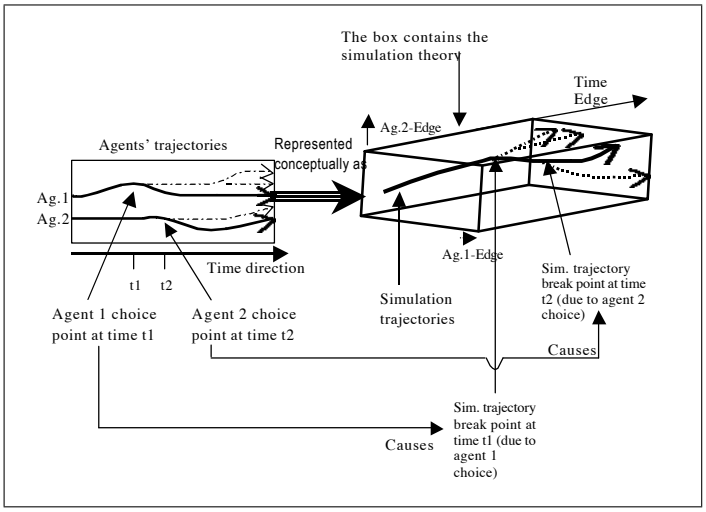


Figure 1.2.14 – *Représentation d'une théorie de simulation sous forme de trajectoire de simulation en prenant en compte les choix des agents. L'espace des trajectoires du système est construit par le produit cartésien de tous les espaces de choix des agents et par la dimension du temps. Une trajectoire dans cet espace spécifie un scénario qui sera utilisé pour la simulation (figure tirée de [TEW]).*

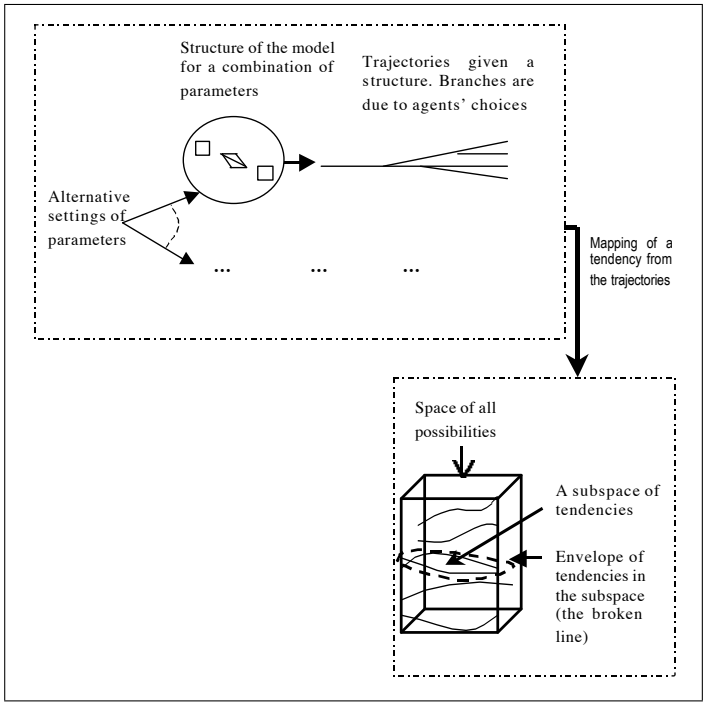


Figure 1.2.15 – *Les trajectoires générées sont fournies au système lors de la simulation pour dégager l'enveloppe des tendances du comportement du système (figure tirée de [TEW]).*

D'autres approches similaires où la notion de morphologie est davantage présente seront présentées plus loin, dans la partie morphologie du manuscrit (chapitre 2).

1.3 Question du contrôle des systèmes multi-agents

Et que peut-on dire du contrôle dans toutes ces méthodes ?

Ce bref exposé sur les systèmes multi-agents nous a permis de voir que les recherches dans ce domaine, du moins celles s'intéressant à la construction et à la validation de tels systèmes, se concentrent sur les méthodes de type génie logiciel et les méthodes d'analyse (vérification et observation), mais que le problème du contrôle n'est pas vraiment abordé.

Selon certains auteurs , la question du contrôle d'un système multi-agents peut même paraître antinomique avec l'esprit de l'agentification. Certains posent que par nature un système multi-agent doit s'autoréguler et donc qu'il n'est point besoin d'en faire le contrôle explicite. Beaucoup d'auteurs aspirent à trouver des mécanismes qui rendent un système multi-agent localement auto-adaptatif. Certains proposent des systèmes qui présentent cette qualité, mais pour des systèmes relativement simples, ou du moins répondant à une tâche bien précise.

À l'heure actuelle il n'y a pas de théorie, ni de méthode satisfaisante permettant de construire des systèmes multi-agents localement auto-adaptatifs et réellement complexes. Au lieu de tenter d'en proposer une, nous adoptons une approche plus humble, et sans doute plus réaliste, empruntant une voie expérimentaliste. Nous pensons que la compréhension de systèmes complexes doit d'abord passer par leur observation et leur analyse, pendant leur fonctionnement. Si l'on dispose d'une méthode générale d'analyse de tels systèmes, nous pouvons espérer savoir – après analyse de différents systèmes – quelles en sont les propriétés communes, dégager des points communs et éventuellement étayer des pistes pour une théorie générale.

Les méthodes orientées génie logiciel donnent l'impression, surtout par le ton de leur présentation, qu'il suffise d'analyser et d'étudier finement le problème *à priori* pour construire un système multi-agents opérant selon le comportement recherché. Nous avons vu que la plupart des méthodes sont semblables à l'esprit d'UML, et d'ailleurs certaines d'entre elles sont des extensions de UML comprenant quelques concepts en plus pour prendre en compte certains aspects des systèmes multi-agents, par exemple la méthode MESSAGE/UML et la méthode ADELFE. Étrangement, beaucoup de ces méthodes passent sous silence, ou « traitent » de façon implicite, le fait que si à la fin du processus conception-réalisation, le système ainsi construit n'opère pas de façon escomptée, qu'il faille réviser le système. Seulement quelques méthodes mentionnent explicitement un besoin éventuel de révision, bien que cela semble « naturel », par exemple dans la méthode Voyelles. Par ailleurs, ces méthodes ne semblent pas prendre en compte les aspects dynamiques et émergents inhérents aux systèmes multi-agents. Peut être que cela n'est pas l'objet de ces méthodes qui s'intéressent uniquement à l'implantation des agents ? Et dans ce cas, quel avantages décisifs apportent ces méthodes par rapport à UML, si ce n'est qu'elles exploitent un vocabulaire anthropomorphe qui peut faciliter les analogies et donc la conception ? Pour de nouveaux problèmes, de nouveaux outils semblent nécessaires.

Il y a cependant des méthodes qui vont plus loin que la simple réalisation et qui

s'intéressent au comportement : ce sont les méthodes d'observation. Nous avons vu qu'il y a les méthodes expérimentales et les méthodes formelles. Néanmoins ces approches se limitent à la vérification et à l'analyse. Elle ne s'intéressent pas au contrôle. Et par ailleurs, la méthode que ces approches proposent ne se prête pas facilement à un mécanisme de contrôle.

Une voie envisageable pour construire des systèmes complexes, du type systèmes multi-agents, est de réaliser un système dont le comportement est approximativement correct, et de moduler ce comportement par le contrôle. Ce sera la voie que nous suivrons.

Chapitre 2

Morphologie

Dans ce chapitre la notion de morphologie, au sens large, sera présentée en s'appuyant sur quelques exemples. Il s'agit d'un bref aperçu dont l'objectif est de permettre d'avoir une idée générale de la notion, de son champ d'application, en donnant quelques références importantes. Tous les exemples remarquables à propos de la morphologie ne sont malheureusement pas évoqués ici. La notion de stabilité structurelle sera présentée en premier lieu (section 2.1). Ensuite différents exemples de morphologie seront mentionnés, en Mathématiques (section 2.2.1), en Biologie (section 2.2.2), en Physique (section 2.2.3), en Informatique (section 2.2.4) et finalement dans le domaine des systèmes multi-agents (section 2.2.4).

2.1 Notion de morphologie et stabilité structurelle

Pour René Thom [Tho72], l'univers n'est pas un chaos, il est peuplé d'objets dont les formes sont récurrentes et qu'il nous est possible de désigner par des mots. Ces formes, pourvues d'une dénomination, sont nécessairement stables. En effet, R. Thom stipule qu'un objet donné naturellement est toujours soumis à des influences perturbatrices de la part du milieu extérieur qui, si faibles soient-elles, auront un effet sur sa forme. Cependant, pour rester reconnaissable aux objets similaires de sa classe, ces perturbations ne devront pas trop changer la forme de l'objet en question.

Dans [Tho72], René Thom donne une définition de la morphologie d'un processus, que nous reprenons ici :

> Un phénomène naturel, sur la nature duquel nous ne faisons aucune hypothèse, se déroule dans une boîte B ; il y aura donc lieu de considérer le produit $B \times T$ de B par le temps T ; ce sera le domaine siège du processus considéré. On admet que l'observateur dispose de sondes ou d'autres moyens qui lui permettent de voir ce qui se passe au voisinage de tout point x de $B \times T$. Une première classification des points de $B \times T$ s'offre d'emblée ; lorsqu'au voisinage d'un point x de $B \times T$, l'observateur ne peut rien déceler de remarquable ; si, phénoménologiquement, le point x ne diffère en rien des points voisins, on dira que ce point x est un *point régulier* du processus. Par définition même, les points réguliers forment un ouvert dans $B \times T$; le complémentaire fermé des points réguliers dans $B \times T$ sera appelé l'ensemble des points *catastrophiques du processus* ; au voisinage de tout point c de l'ensemble fermé K des points catastrophiques, le processus présente une discontinuité : « il se passe quelque chose » dans toute boule de centre c. La donnée du fermé K – et la description des singularités en chacun de ces points – constitue la *morphologie* du processus.

En termes moins techniques, nous pouvons dire que R. Thom choisit de considérer un phénomène comme une morphologie, c'est à dire comme un système de discontinuités qualitatives sur un espace substrat.

En empruntant un vocabulaire mathématique, on peut dire [Tho72] qu'il existe donc dans un espace E un ouvert constitué des points représentant des formes structurellement stables, le complémentaire de cet ouvert comportant seulement des formes instables, qu'une infime perturbation peut transformer.

Intuitivement parlant, une situation est structurellement stable quand les situations voisines lui ressemblent. L'importance de cette notion est donc évidente, un phénomène structurellement instable ayant peu de chances d'être observé de manière stable. D'un point de vue expérimental, l'idée de la stabilité structurelle est que si l'on peut répéter une expérience approximativement dans les mêmes conditions, on obtiendra approximativement les mêmes résultats.

2.2 Exemples

Dans cette section nous passerons en revue quelques exemples d'applications de la notion de morphologie dans des domaines relativement divers. Cet exposé n'est pas exhaustif, son but est seulement de donner quelques points de repère.

2.2.1 En Mathématiques

René Thom est un mathématicien qui s'est intéressé, entre autre, à la modélisation des processus de développement d'organismes biologiques. Son ouvrage « Stabilité structurelle et morphogénèse » [Tho72] cherche à donner un certain formalisme qui permet d'attaquer le problème de morphogénèse de façon générale. Ce formalisme, issu d'une réflexion sur les mécanismes à l'œuvre dans le développement embryologique, débouche sur une méthode universelle permettant d'associer à toute apparence morphologique la situation dynamique locale qui l'engendre, et ceci de manière indépendant du substrat qui en est le support, qu'il soit matériel ou non.

René Thom a ainsi développé la fameuse théorie des catastrophes élémentaires, qui permet de représenter l'ensemble des discontinuités, nommées « catastrophes », d'un système dans des cas simples (fonctions lisses à variables réelles) sur l'espace-temps $\mathbb{R}^4$. Ces discontinuités sont rapportées à des bifurcations (disjonction ou réunion) d'attracteurs.

Un attracteur est défini comme suit [Tho72] :

Soit (M,X) un système dynamique défini par le champ de vecteurs X dans la variété M. Un attracteur F de ce système est un ensemble fermé invariant par X doué des trois propriétés suivantes :

- il existe un ouvert U, voisinage ouvert invariant de F, le bassin de l'attracteur F, tel que toute trajectoire issue d'un point de U admet F comme ensemble ω-limite ;
- toute trajectoire, dont l'ensemble α-limite contient un point de F, est dans F ;
- F est indécomposable, c'est-à-dire que presque toute trajectoire de X dans

F est dans dense F_1.

Un potentiel est une fonction scalaire prenant une valeur en tout point de l'espace caractéristique d'un champ de force conservatif. En effet, toutes les forces conservatives dérivent d'un potentiel. Une force est dite conservative lorsque le travail produit par cette force est indépendant du chemin suivi par son point d'action, un exemple est la force gravitationnelle. Ainsi, si l'on note la force conservative f et le potentiel V :

$$f = -gradV$$

On peut faire le tableau de toutes les singularités du potentiel V qui se présentent de manière structurellement stable sur $\mathbb{R}^4$, et donner le modèle algébrique correspondant des surfaces de catastrophe. À titre indicatif, en voici la liste :

Nom	Équation
Pli	$x^3 + ux$
Ombilic hyperbolique	$x^3 + y^3 + uxy + vx + wy$
Fronce	$x^4 + ux^2 + vx$
Queue d'aronde	$x^5 + ux^3 + vx^2 + wx$
Papillon	$x^6 + ux^4 + vx^3 + wx^2 + tx$
Ombilic elliptique	$\frac{1}{3}x^3 - xy^2 + u(x^2 + y^2) + vx + wy$
Ombilic parabolique	$x^2y + y^4 + ux^2 + vy^2 + wx + ty$

2.2.2 En biologie

La notion de morphologie est plus connue sous le terme de morphométrie en Biologie, sous sa forme moderne.

D'Arcy Thompson

Un des précurseurs de la notion de morphologie fut d'Arcy Thompson, qui a en 1917 écrit une œuvre de référence « On Growth and Form » [Tho17]. D'Arcy Thompson était particulièrement intéressé par l'unité de la vie et la relation entre

la forme et la fonction. Ses nombreux dessins et croquis reflètent non seulement son amour pour la nature – les coquillages, les grenouilles, les flocons de neiges et les bulles... – mais révèlent aussi sa perception de motifs mathématiques et mécaniques, qui était bien en avance sur son temps. Une des critiques à l'égard de son travail était que ses résultats ne comportaient pas vraiment d'analyses statistiques, symbole de la science à l'époque. Thompson recherchait plutôt l'uniformité que la différence. Comme il le disait, la recherche des différences ou des contrastes fondamentaux entre les phénomènes organiques et inorganiques ou des choses animées et inanimées, a occupé beaucoup d'esprits alors que la recherche des principes communs et les similarités essentielles est faite par peu de chercheurs. Malgré l'apparente diversité des formes, il n'y a pas tant de possibilités de se conformer, et d'Arcy Thomspon l'avait bien remarqué.

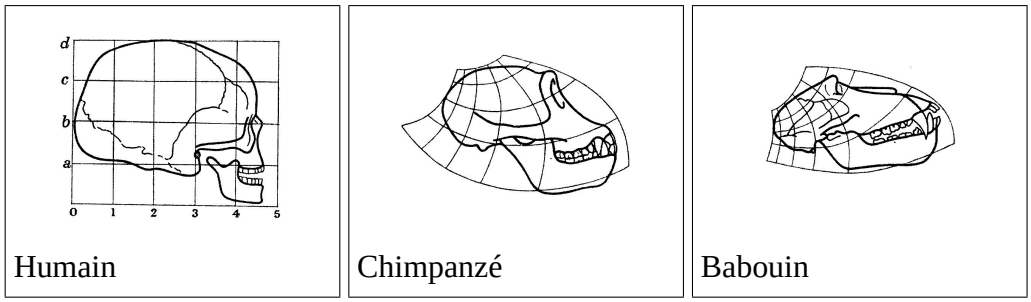


Figure 2.2.1 – Morphologie de crâne d'espèces différentes (d'après d'Arcy Thompson, images tirées de [Tho17]).

En termes plus actuels, nous pouvons dire qu'il s'intéressait à la stabilité structurelle du phénomène de croissance des organismes : la recherche de motifs communs dans les formes des espèces différentes. La ressemblance des organismes, par exemple, illustrée sur la figure 2.2.1, pourtant d'espèces différentes, reflète bien cette notion de stabilité de ces processus physiques et biologiques.

Morphométrie

Les méthodes de mesures et de calculs étant devenus plus sophistiquées, l'étude de la forme des organismes devient alors davantage quantitative et précise, mais en restant néanmoins toujours dans la lignée des travaux de d'Arcy Thompson. L'analyse de la morphologie [Boo91] des organismes est une donnée fondamentale en paléontologie (reconstitution de l'histoire de la Terre à partir des vestiges

du passé), car il s'agit souvent de la seule information disponible dans le registre fossile, susceptible de fournir des indications sur la paléobiologie (étude des relations évolutionnaires et écologiques entre les organismes fossiles et les paléoenvironnements dans lesquels ils ont vécu) et l'évolution des organismes. Les approches quantitatives sont longtemps restées limitées à des variables de taille (distances) ou d'angles. Le développement récent de la morphométrie géométrique permet la quantification des variations de forme.

Les types de mesures effectuées peuvent être par exemple :

- des mesures de paramètres de taille (distances, surfaces) et d'angles ;
- l'analyse des points de repères : chaque objet est représenté par une configuration de points de repères, c'est-à-dire un ensemble de coordonnées x et y, sélectionnés par un opérateur (la personne qui mène l'analyse). Les individus étudiés sont ensuite superposés au terme d'une standardisation de la taille, suivie d'étapes de translation et de rotation. La comparaison de ces différentes configurations permet l'analyse de la variabilité de forme, qui peut être visualisée sous la forme de grille de déformation ;
- l'analyse de contours : chaque objet est décrit par un ensemble de points échantillonnés le long du contour. À partir de ces données, une fonction empirique du contour sera considérée, dépendant de la méthode choisie (variation du rayon, des coordonnées x et y, etc... en fonction de l'angle au centre, de la distance cumulée le long du contour, etc.). Au moyen de l'analyse de Fourier, on peut ajuster à cette fonction empirique une fonction mathématique correspondant à la somme de fonctions trigonométriques (les harmoniques) de longueur d'onde décroissante. À chaque harmonique correspondent des coefficients de Fourier, qui pondèrent les fonctions trigonométriques. L'ensemble des coefficients de Fourier des différentes harmoniques décrit l'objet initial d'une manière comparable entre individus. Ils peuvent être analysés par des méthodes de statistiques multi-variées ;
- l'analyse fractale : cette méthode permet d'estimer le degré de complexité morphologique d'un objet et fournit une valeur (la dimension fractale) comparable entre spécimens.

Exemple modèle proie-prédateur

Dans cet exemple, nous exposons la notion d'espace des phases, où les idées de stabilité structurelle et de morphologie sont bien présentes. Nous allons considérer donc ici un système « proie-prédateur ». L'un des premiers modèles à incorporer l'interaction entre deux espèces (proie et prédateur) fut proposé en 1925 par Alfred Lotka et Vito Volterra pour la modélisation des fluctuations des populations de gros poissons (prédateurs) et de petits poissons (proies) [BK99].

Le modèle Lotka-Volterra est composé d'une paire d'équations différentielles qui décrit la dynamique entre les proies et les prédateurs. Ces deux équations peuvent s'écrire :

$$\frac{dp}{dt} = rp - apP$$

$$\frac{dP}{dt} = bpP - mP$$

où :

- p : densité des proies ;
- P : densité des prédateurs ;
- r : taux de croissance intrinsèque des proies ;
- a : coefficient de prédation ;
- b : taux de reproduction des prédateurs par proie mangée ;
- m : taux de mortalité des prédateurs.

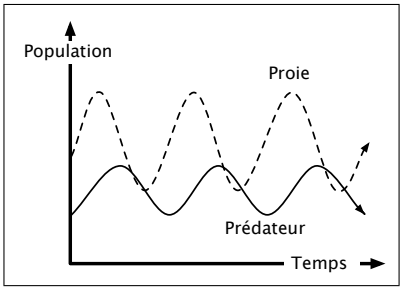


Figure 2.2.2 – *Schéma rendant compte des fluctuations temporelles des populations de gros et petits poissons constatées dans les données statistiques sur mesures.*

Ce modèle se caractérise par des oscillations de la taille de la population des deux espèces, proies et prédateurs, avec les pics de population des prédateurs un peu retardés par rapport à ceux des proies (voir figure 2.2.2).

Ce modèle avait des hypothèses simplificatrices, qui se reflètent dans les équations :

- la population des proies augmentent exponentiellement en l'absence de prédateurs ;
- la population des prédateurs est affamée en l'absence de proie (au lieu d'aller chasser d'autres proies) ;
- les prédateurs peuvent manger un nombre infini de proies ;
- l'environnement n'est pas pris en compte, il n'y a pas de dynamique de l'environnement.

La figure 2.2.2 illustre l'évolution des deux populations selon l'axe du temps. On peut s'intéresser à l'évolution des deux populations en fonction de l'une et de l'autre. Ceci est représenté dans l'espace des phases à deux dimensions, dont la coordonnée sur chacun des axes représente la taille de la population. Schématiquement, cela est représenté sur la figure 2.2.3. Ce schéma ne tient pas compte exactement du décalage de phase qu'il peut y avoir entre les deux courbes temporelles de populations.

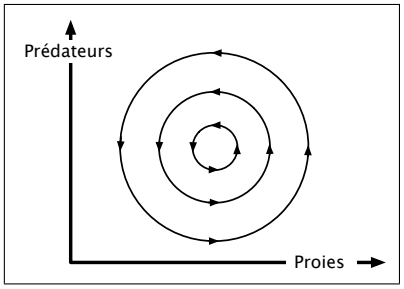


Figure 2.2.3 – *L'espace des phases du modèle simple.*

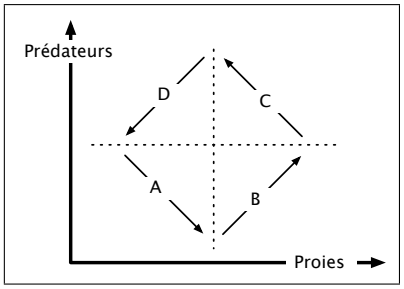


Figure 2.2.4 – *Modèle proie-prédateur : espace des phases.*

En analysant la figure 2.2.3, on peut caractériser quatre tendances du système, qui sont reprises sur la figure 2.2.4 :

- région A : dans cette zone de l'espace des phases, les deux populations sont relativement petites, les prédateurs diminuent en nombre par manque de nourriture (les proies) tandis que ces derniers augmentent grâce à la diminution de la prédation ;
- région B : dans cette région, il y a beaucoup de proies et relativement peu de prédateurs. Lorsqu'il y a beaucoup de proies et de prédateurs, les deux populations augmentent ;
- région C : dans ce cas, les deux populations sont relativement importantes. Les prédateurs se nourrissent bien et se multiplient, ainsi la population des proies décline ;
- région D : il y a peu de proies et beaucoup de prédateurs. Les deux populations déclinent simultanément.

En incorporant un phénomène de « friction » dans la dynamique entre les deux

populations, modélisant par exemple le fait que les populations sont de moins en moins fertiles (moins de reproduction), le système proie-prédateur tendrait vers un équilibre, comme cela est illustré sur la figure 2.2.5.

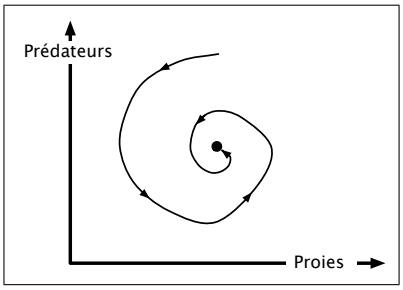


Figure 2.2.5 – *Si l'on introduit une sorte de « friction » écologique dans le système, le modèle produirait des trajectoires qui convergeraient vers un point fixe. Ce type de modèle serait acceptable pour des modélisations à équilibre statique.*

Un modèle peut-être un peu plus adéquat, incorporant le phénomène de friction et de gêne mutuelle entre les espèces (par exemple, plus il y a de proie plus elles ont du mal à s'échapper dans le même environnement. Et respectivement, plus il y a de prédateurs plus ils se gênent pour chasser). Un tel modèle pourrait se traduire par un portrait de phase illustrer sur la figure 2.2.6.

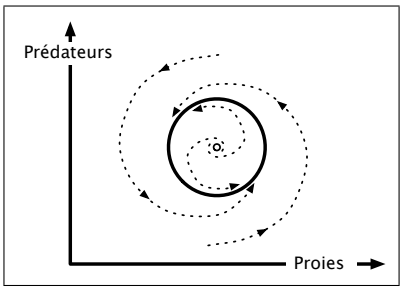


Figure 2.2.6 – *Modèle plus adéquat qui rendrait mieux compte des variations constatées sur les données relevées.*

2.2.3 En physique

Exemple du pendule

Nous allons présenter un autre exemple d'espace des phases en physique pour un système simple. Nous considérerons le cas du pendule. Un pendule est un système composé d'une masse m attachée à un fil inextensible de longueur L (voir figure 2.2.7). Nous allons considérer son mouvement dans le plan pour simplifier l'étude. Les hypothèses sont : masse constante, fil inextensible, poids du fil négligé, seule la gravité g intervient sur sur système dans un premier temps, ensuite nous tiendrons compte des frottements (dus à l'air par exemple).

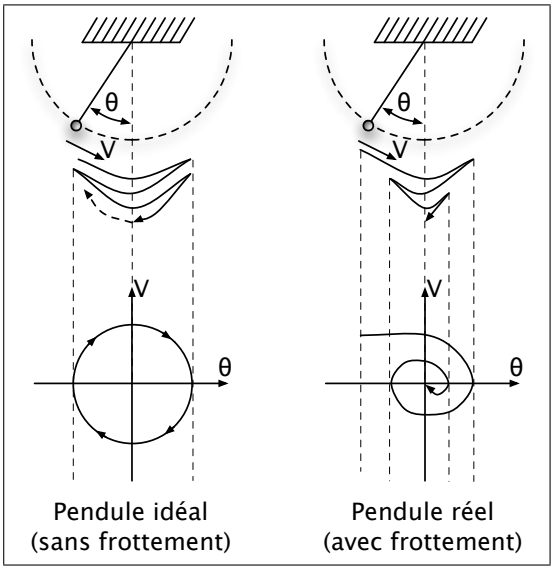


Figure 2.2.7 – *Schéma de pendules et des espaces des phases correspondants.*

En schématisant un peu, nous pouvons représenter l'état de ce système (le pendule) par les deux grandeurs suivantes :

- la position angulaire θ de la masse par rapport à la position d'équilibre ;
- la composante tangentielle V de la vitesse de la masse.

Nous allons considérer deux cas : un cas hypothétique du pendule dit « idéal » (sans frottement) et le pendule réel (avec frottement).

Dans le cas du pendule idéal, on fera abstraction des frottements. Le système adopte un comportement périodique. Cela se traduit par une courbe fermée dans l'espace des phases. Le fait que le portrait du système dans l'espace des phases est une courbe fermée reflète que le comportement du système est périodique : le pendule oscille toujours de la même façon (voir la figure 2.2.7, partie de gauche).

Dans le second cas, nous introduisons les effets des frottements qui se traduisent par dissipation de l'énergie mécanique ; ce cas est un peu plus réaliste que le premier cas. Dans ce cas le pendule oscillera avec une amplitude décroissante pour finir dans un état d'équilibre statique : le pendule finira par ne plus osciller car il aura dissipé toute son énergie dans les frottements. Cela se traduit dans l'espace des phases par une trajectoire qui converge vers le point fixe $(V, \theta) = (0, 0)$. Et ce quel que soit le point de départ du système (voir la figure 2.2.7, partie de droite).

Exemple diagramme d'état de l'eau

Nous allons considérer un autre système physique très simple : un certain volume d'eau. Nous allons prendre en compte deux paramètres de ce système :

1. sa pression ;
2. sa température.

L'eau peut exister sous trois états :

– l'état solide (la glace) ;
– l'état liquide (l'eau) ;
– l'état gazeux (la vapeur).

Le passage de l'un à l'autre de ces états nécessite un apport ou une perte d'énergie et porte un nom spécifique. On utilise dans le langage technique le terme de condensation pour désigner le passage de la vapeur au liquide.

État de départ	État de d'arrivée	Nom de la transition	Nom de la transition (sens inverse)
solide	liquide	fusion	solidification
solide	vapeur	sublimation	condensation
liquide	vapeur	vaporisation	liquéfaction

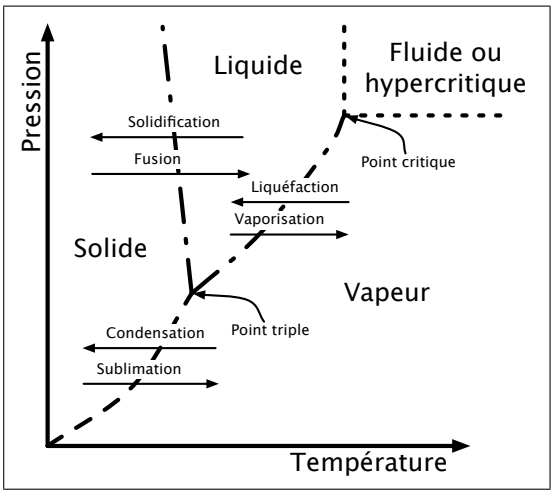


Figure 2.2.8 – *Diagramme de l'eau.*

Si l'on fait une expérience qui consiste à changer les paramètres d'un volume d'eau, sa pression et sa température, et que l'on observe les différents états de ce volume d'eau, aux différentes valeurs de pressions et température, et que l'on reporte sur un graphique à deux dimensions, dont les axes sont la température et la pression, on obtiendrait un graphe similaire à celui de la figure 2.2.8. On pourrait procéder de même avec d'autres types de matériaux, par exemple du métal. et on obtiendrait sensiblement la même forme de graphique, les différentes transitions seraient placées de manière un peu différente que dans le cas de l'eau. Mais néanmoins la forme globale du graphe serait la même.

Pour une température donnée, le passage d'un état à l'autre ne peut se faire qu'à une pression fonction de cette température (et inversement). Dans le cas de l'eau, le point triple, point précisant la pression et la température pour lesquelles on peut avoir coexistence des trois états, est situé à la température de 0,01 °C et une pression de 611 Pa. Au dessus du point critique (374 °C et 221 kPa) il n'y a plus de différence entre l'état liquide et l'état gazeux : on désigne ce fluide par le terme d'hypercritique.

Un comportement intéressant de l'eau, par rapport à d'autres matériaux, est celui de la transition solide-liquide. En effet si l'on prend de la glace et qu'on augmente la pression, on voit qu'elle va fondre. C'est ainsi que sous la pression d'un patin sur la glace celle-ci va fondre et les patineurs évoluer sur un film d'eau liquide, l'eau se solidifiant à nouveau derrière le patin. En effet, cela se remarque sur le diagramme par la séparation solide/liquide qui est une pente négative, si l'on augmente suffisamment la pression, on passe de l'état solide à l'état liquide. Pour d'autres matériaux la pente est positive.

Bien qu'il ne s'agisse pas véritablement d'un espace des phases, cette représentation permet de mettre en évidence une relation entre l'état de l'eau et deux des ses propriétés. On peut dire que ce diagramme représente un certain modèle de l'eau.

Contrôle de systèmes chaotiques

En Physique, de façon très générale, nous pouvons distinguer deux sortes de systèmes : les systèmes linéaires et les systèmes non-linéaires. Les premiers peuvent être décrits par des équations linéaires, par exemple $\frac{\delta e}{\delta t} = f(e)$. Supposons que nous trouvons deux solutions $e_1(t)$ et $e_2(t)$. La somme de ces deux solutions $e_1 + e_2$ est également solution selon le principe de superposition. Les solutions d'une équation linéaire forment un espace vectoriel, ce qui permet l'utilisation de l'algèbre linéaire et simplifie considérablement l'analyse de tels systèmes. En outre, pour les systèmes à temps continu, la transformée de Laplace permet de transformer les équations différentielles en des équations algébriques. En revanche, les systèmes non-linéaires ne répondent pas à ces critères. Leur analyse est en général très difficile.

Les systèmes que l'on appelle « chaotiques » sont des systèmes non-linéaires. Il sont caractérisés par la propriété d'avoir un comportement apparemment désordonné. Même si les lois régissant leur comportement sont connues et simples – par exemple dans le cas de la fonction dite de logistique – il se peut que la prévision soit impossible à moyen et long terme.

Pour illustrer la notion de comportement chaotique, considérons la fonction de logistique, qui est définie de la manière suivante :

$$f_a : x \mapsto ax(1-x)$$

On considère alors la suite :

$$u_{n+1} = f_a(u_n)$$

Selon les différentes valeur du paramètre a, la suite u_n présente des propriétés différentes. Différentes observations possibles sont résumées ci-dessous (voir figure 2.2.9) :

- pour $a = 1,5$: la suite u_n converge rapidement de façon monotone ;
- pour $a = 2$: la suite u_n converge rapidement de façon monotone également ;
- pour $a = 2,5$: la suite u_n converge rapidement, après un premier pic ;
- pour $a = 3$: la suite commence à osciller autour, et ce de façon périodique ;
- pour $a = 3,5$: la suite oscille, mais la période est de 4 ;
- pour $a = 4$: la suite a un comportement apparemment non prévisible, on appelle un tel comportement un comportement « chaotique » ;

Tant que le paramètre a est compris entre 0 et 4, l'intervalle $[0,1]$ est stable et l'on peut observer les différentes itérations. Le « déterminisme » est assuré pour les valeurs de a proche de 0, le « chaos » apparaît lorsqu'il croît.

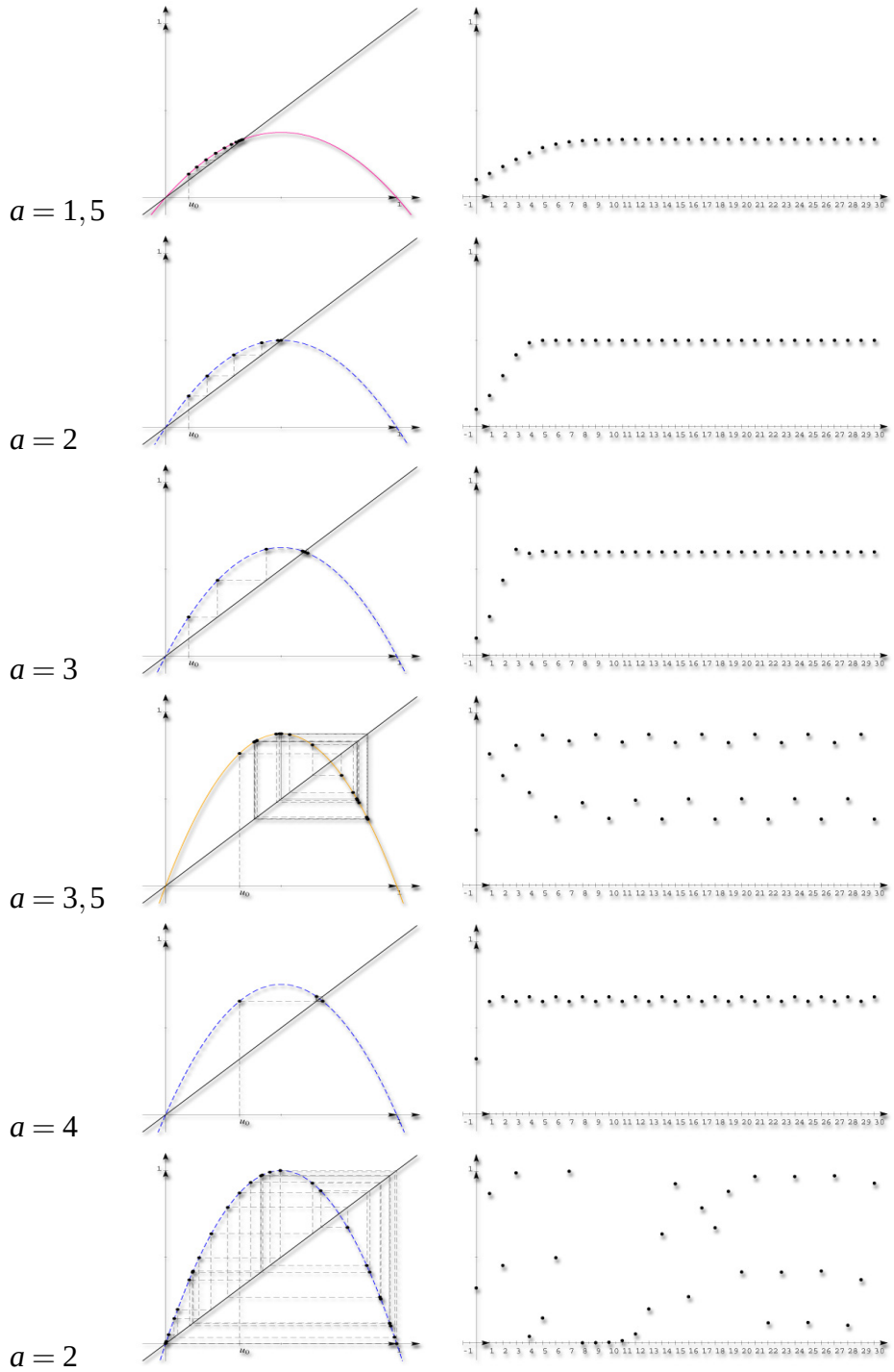


Figure 2.2.9 – Différents comportement de la fonction logistique en fonction de différentes valeurs du paramètre a.

Une autre caractéristique des systèmes chaotiques est leur sensibilité aux conditions initiales. Un petit écart peut entraîner des grandes différences très rapidement. Nous itérons la fonction logistique $f_{0,4}$ ($a = 0,4$ dix fois pour des valeurs initiales sensiblement pareilles 0.4 et 0.4001. Au bout de la dixième itération, l'écart (en valeur absolue) entre les deux suites est de l'ordre de 0.099. Nous illustrons cela par le tableau suivant.

n	$x_0 = 0.4$	$x_0 = 0.4001$	$x_n - x'_n$
0	0.4	0.4001	-0.0001
1	0.96	0.96007996	0.00007996
2	0.1536	0.153305721626	0.000294278374406
3	0.52002816	0.51921230937	0.000815850630201
4	0.998395491228	0.998523548675	-0.000128057446659
5	0.00640773729417	0.00589708566707	0.000510651627103
6	0.0254667127878	0.0234492401908	0.00201747259695
7	0.0992726373102	0.0915974933012	0.00767514400903
8	0.357670323167	0.332829570088	0.0248407530783
9	0.91896905237	0.888216189453	0.0307528629174
10	0.297859732624	0.397152760987	-0.0992930283628

Nous venons de voir que, même pour un système très simple, tel que l'itération de la fonction de logistique, il se peut que la prévision soit impossible à moyen et long terme. C'est une des caractéristiques de ces systèmes : ils sont très sensibles aux conditions initiales. Cela se traduit, dans l'espace des phases, par le fait que deux trajectoires infiniment proches au départ, tendent à diverger de façon importante au cours du temps.

Les physiciens utilisent, entre autre, l'exposant de Lyapunov pour caractériser le comportement chaotique ou non des systèmes [Hil94]. L'exposant de Lyapunov est une mesure de la vitesse d'attraction ou de répulsion d'un point fixe dans l'espace des phases. Cette notion peut également être appliquée à la divergence de trajectoires proches de tout point en général dans l'espace des phases.

Les systèmes présentant de tels comportements sont relativement répandus dans la nature. Leur étude présente de l'intérêt du point de vue de leur contrôle. Un des intérêts des systèmes non linéaires est que, bien que le contrôle y est peut être plus difficile, il est intéressant de découvrir pourquoi de petits effets peuvent avoir de grandes conséquences.

Bien qu'il soit difficile de faire de la prévision formelle ou exacte de tels sys-

tème, il est néanmoins possible de faire de la prédiction qualitative, qui peut être fort utile. La prévision qualitative se fait en général par l'étude du système dans un espace des phases. La projection du comportement d'un système chaotique dans un espace des phases permet, souvent, de révéler des attracteurs de comportement du système.

Même si ces systèmes se prêtent peu à la prévision exacte, ils ont néanmoins intéressés les chercheurs, notamment en physique, où ces derniers ont développé des techniques d'analyse et de contrôle. Un des intérêt de pouvoir contrôler de tels systèmes non linéaires est qu'un moindre effet sur le système peut avoir une influence importante sur son comportement [OGY90].

Un des deux ingrédients majeurs [Sch99] pour le contrôle de systèmes chaotiques passe par l'observation que l'ensemble chaotique, dans lequel les trajectoires d'un système chaotique résident, comporte de nombreuses trajectoires à petite période instables. Certaines de ces trajectoires peuvent correspondre à un comportement souhaité du système. Le second ingrédient est de réaliser que, bien que « chaotique » signifie que de petites différences dans l'état du système entraîne de grande divergence dans le comportement à terme, cela implique également que le comportement du système peut être modifié par de petites perturbations.

La méthode OGY [OGY90] (baptisée du nom de ses auteurs : Ott, Grebogi, Yorke) propose de stabiliser des orbites périodiques instables dans un attracteur chaotique par des perturbations dans le temps de paramètres de contrôle. La méthode OGY peut être décrite en trois étapes :

1. déterminer les orbites instables à petites périodes dans l'attracteur chaotique ;
2. examiner la position et la stabilité de ces orbites ;
3. appliquer de petites contraintes sur le système pour stabiliser les orbites choisies.

Cette méthode est applicable même si l'on de dispose pas de modèle global du système. Cette analyse peut être effectuée à partir de données expérimentales par analyse non linéaire de séries temporelles. Ceci est important car, en général, les systèmes chaotiques sont plutôt compliqués et les équations régissant le processus sont souvent inconnues. De nombreuses applications théoriques ou pratiques peuvent être trouvées dans [Sch99].

2.2.4 En informatique

Attracteurs de générateurs de nombres aléatoires

Nous présentons à présent un exemple d'étude de formes appliqué à l'informatique.

Michal Zalewski [Zal02] s'est intéressé au problème de sécurité réseau suivant : l'introduction de paquets malicieux dans une connexion TCP, et celui de l'établissement d'une connexion en utilisant une adresse légitiment utilisé par une autre machine.

Lors d'une connexion TCP/IP [Ste] à un hôte, ce dernier génère un numéro initial de séquence « ISN » (« Initial Sequence Number »). Ce numéro de séquence est utilisé pour la conversation entre les deux machines. Dès les années 1980, on avait émis l'idée que l'on pouvait s'appuyer sur ce numéro de séquence pour détourner une connexion et y insérer des paquets. Les implémenteurs de piles TCP/IP doivent donc bien prendre garde à ce que la séquence des numéros de séquence ne soit pas facilement prévisible.

En fait, cette étude de faisabilité d'attaque sur système informatique dans ce cas revient à l'étude de la prévisibilité de générateurs des nombres pseudo-aléatoires utilisés dans les piles TCP/IP. Néanmoins le rapprochement entre les deux domaines est intéressant.

La génération de nombres aléatoires par ordinateur n'est pas évidente. Ceci est dû au fait que les ordinateurs sont conçus justement pour faire exactement ce qu'on leur dit de faire. Et donc un même algorithme (par exemple, de générateur de nombres aléatoires) produira les mêmes résultats sur différentes machines. Même si le générateur de nombres pseudo-aléatoires n'est pas connu, cet algorithme générera la même séquence de nombres du fait que la machine et l'algorithme possèdent un nombre d'états fini. Fort heureusement cela prend en général du temps (de l'ordre du mois ou de l'année). Mais il arrive que certaines implantations de générateurs de nombres aléatoires bouclent au bout de quelques centaines d'itérations. La solution à ce problème est de rajouter de l'entropie dans les générateurs. ceci est techniquement fait en prenant en compte certains paramètres imprévisibles, tels que les touches appuyées sur un clavier, l'état de certaines interruptions entrées/sorties, etc...

Le numéro de séquence tient sur 32 bits. Donc, théoriquement, si un attaquant génère les 2^{32} (4 294 967 296) paquets contenant chacun un numéro de séquence différent, l'attaque réussira. Cependant cette approche n'est pas envisageable, car elle prendrait trop de temps et de bande passante. L'idée est alors de réduire la taille de l'ensemble des paquets à essayer. M. Zalewski introduit l'idée de « ensemble trompeur » (« spoofing set »). Il s'agit d'un ensemble de valeurs ISN devinées utilisées pour faire une inondation de paquets afin de corrompre la connexion, le but étant de réduire la taille de cet ensemble pour rendre l'attaque faisable.

Afin de produire un ensemble trompeur, M. Zalewski passe par une analyse morphologique des générateurs de nombres aléatoires. Pour cela, il utilise une technique appelée « méthode du retard » (ou « coordonnées retardées »). Si l'on a une séquence temporelle s_n, on construit la représentation graphique en calculant les points p_n, par exemple dans un espace à trois dimensions, de coordonnées (x_n, y_n, z_n) définis de la façon suivante :

$$x_n = s_{n-2} - s_{n-3}$$

$$y_n = s_{n-1} - s_{n-2}$$

$$z_n = s_n - s_{n-1}$$

En appliquant cette technique aux séquence ISN, Zalewski a construit expérimentalement les attracteurs de ces séquences pour différents systèmes d'exploitation, voir la figure 2.2.10.

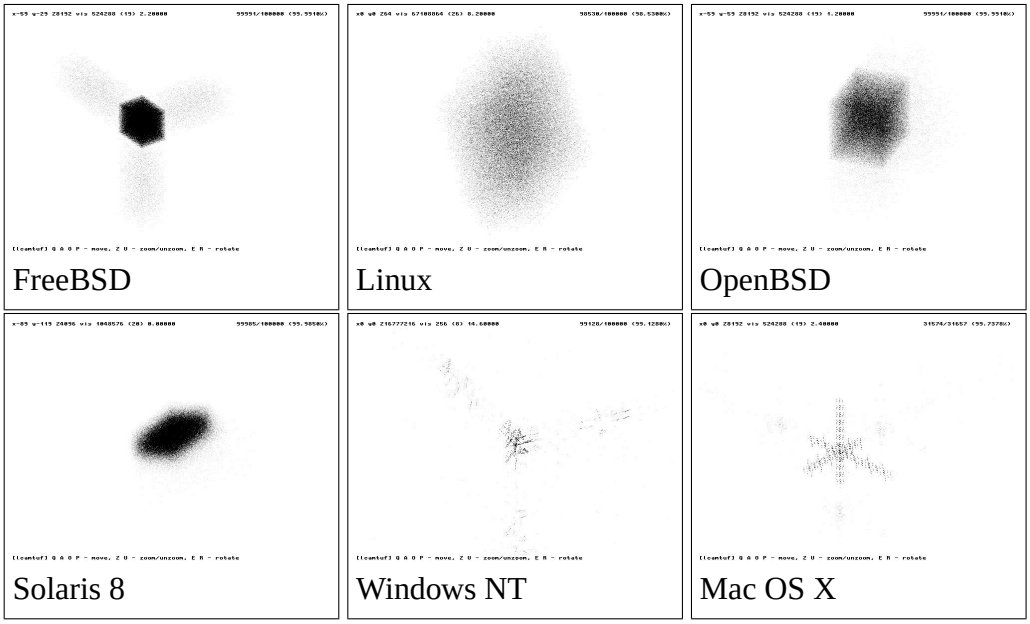


Figure 2.2.10 – *Attracteurs de nombres pseudo-aléatoires utilisés dans les piles TCP/IP de différents systèmes (d'après Michal Zalewski, images tirées de [Zal02]).*

Les figures ci-dessus illustrent « l'attracteur étrange » des générateurs de nombres pseudo-aléatoires utilisés pour les séquences SIN de TCP/IP des systèmes d'exploitation respectifs. L'auteur de ces articles a construit ces attracteurs par la méthode du retard. Il s'agit d'une méthode simple et efficace pour construire une représentation multidimensionnelle d'une séquence de nombres. Cette représentation permet, parfois, de mettre en évidence des structures insoupçonnées dans la séquence numérique.

Ensuite il se fonde sur l'hypothèse suivante : « Si une séquence présente un fort attracteur, alors les valeurs futures de la séquence seront proches des points précédents ayant servis à construire l'attracteur ». Il s'agit là d'une hypothèse relativement raisonnable. Si les attracteurs ont une structure complètement aléatoire, ce qui se traduit par l'absence de forme notable, on ne peut pas prévoir grand chose. En revanche, si les attracteurs possèdent une structure particulière, notamment géométrique avec des symétries, il est alors possibles de faire de la prévision.

Supposons que l'on repère un numéro de séquence s_n par (x,y,z). Dans le

cas où les attracteurs sont significatifs, il est raisonnable de dire que la prochaine valeur du numéro de séquence se trouve quelque part sur la droite D de l'espace $Y = x, Z = y$, ou que le numéro de séquence s_{n+1} est spécifié par : (y, z, w) ou w est une inconnue (voir la figure 2.2.11).

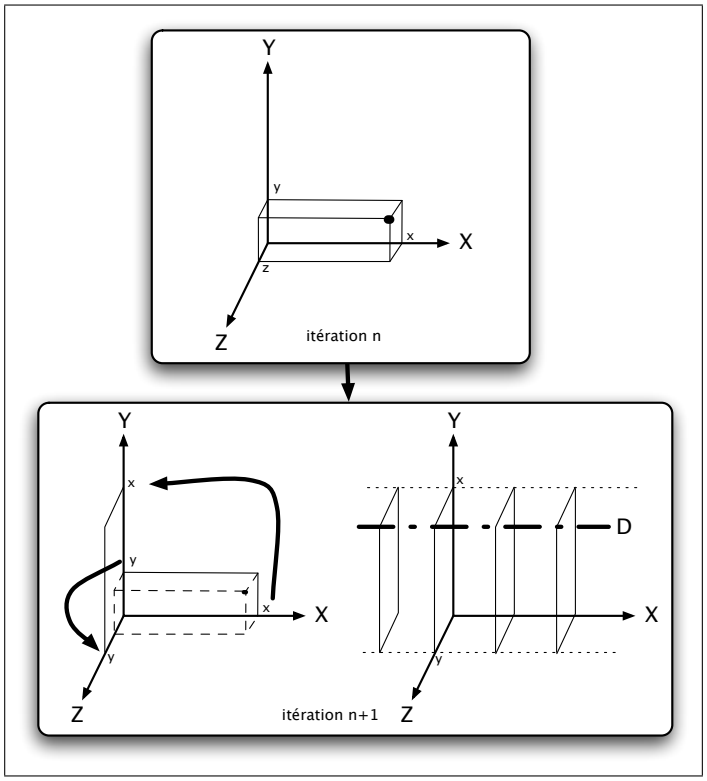


Figure 2.2.11 – *La méthode du retard.*

Les points inclus dans l'ensemble trompeur sont alors :

- les points de l'attracteur sur la droite D ;
- les points au voisinage de la droite D ;
- les points voisins des points ci-dessus.

Par souci de tester des attaques réalisables, M. Zalewski limite la taille de cet ensemble à 5000 numéros ISN.

En suivant cette méthode et des données expérimentales, M. Zalewski a estimé la faisabilité d'attaque fondée sur le numéro de séquence ISN. Quelques résultats

préliminaires sont présentés dans [Zal02].

Dans les systèmes multi-agents

La notion de morphologie est relativement peu présente dans les recherches sur les systèmes multi-agents. Les travaux de Teran sur l'analyse des systèmes multi-agents font intervenir la notion de morphologie par la notion d'enveloppe.

Travaux d'Alain Cardon

Dans [Car99, Car03], Alain Cardon présente de façon générale son modèle de contrôle de systèmes multi-agents fondé sur l'analyse morphologique. Nos propres travaux s'inscrivent clairement dans la continuité des travaux d'Alain Cardon et tentent de fournir un approfondissement et une réalisation concrète d'un tel système multi-agent contrôlé morphologiquement.

Thèse de Franck Lesage

Les travaux réalisés dans le cadre de la thèse de Franck Lesage [Les00] ont consisté dans le développement d'un prototype de système multi-agents réalisant une interprétation « artificielle » des messages échangés entre les acteurs d'un système d'aide à la décision de gestion de crise. Les systèmes d'information et de communication (SIC) sont des systèmes hérités des systèmes de commandement militaires et sont utilisés pour la gestion de situations d'urgence mettant en œuvre des ressources matérielles et humaines. Leurs fonctionnalités sont censées permettre une meilleure prise de décision, en temps réel, à partir des connaissances hautement dynamiques, floues, imprécises et qui ne sont pas toujours fiables car remontant d'intervenants multiples sur le terrain [Les00]. Les travaux de F. Lesage sont une continuité des travaux de thèse de Stéphane Durand [Dur99]. Par ailleurs, Jean-Marie Auvinet a étudié l'applicabilité de ces approches dans le cadre de la police nationale française, pour assister les opérateurs radio des salles d'informations et de commandement [Auv05].

Le système de F. Lesage analyse et interprète des messages échangés entre les

acteurs du système d'information et de communication. La structure globale du système d'interprétation comporte quatre couches (voir figure 2.2.12) :

- une couche aspectuelle qui représente la situation telle qu'elle a été exprimée par les acteurs ;
- une couche morphologique qui permet de dégager les faits significatifs de la couche aspectuelle ;
- une couche d'agents homéostatiques qui assurent une certaine stabilité du système. C'est un élément qui vérifie la cohérence du système, ce qui est une des fonctions importantes d'un outil d'assistance à la gestion de crise ;
- et une couche d'analyse qui traite les faits saillants dégagés par la couche de morphologie.

Le système est par ailleurs interfacé avec l'environnement par une couche communicationnelle asynchrone, permettant l'échange d'information. Chaque aspect de la situation est représenté par un agent aspectuel qui suit l'évolution de la représentation d'un aspect de la situation que se fait un acteur sur la base de ses messages. Un agent aspectuel est une agentification d'un trait sémantique dans le discours entre les acteurs.

Tout message échangé entre des acteurs est d'abord automatiquement transformé en une organisation active d'agents aspectuels, ces agents représentant les caractères sémantiques du texte du message.

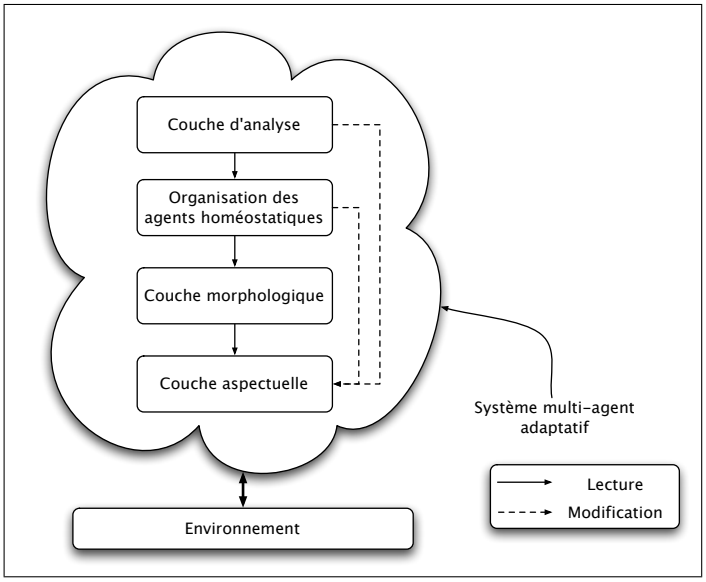


Figure 2.2.12 – *Architecture générale du système proposé par Lesage.*

Le système de F. Lesage repose sur une interprétation artificielle des messages transitant dans le système. Pour construire une représentation de l'état du système d'aide à la décision, en l'occurrence de l'organisation d'agents aspectuels, F. Lesage adopte une approche morphologique. Il propose une construction d'une synthèse géométrique de l'état de l'organisation selon des critères de développements des agents. Pour ce faire, F. Lesage introduit la notion d'agents « pré-morphologique » et d'agents morphologiques qui opèrent sur l'organisation aspectuelle.

F. Lesage avait adopté une approche mécaniste de la morphologie qu'il proposait. Il a envisagé de faire de la prédiction à court et moyen terme sur l'évolution de l'état du système multi-agents. Il avait proposé des définitions de mesures de propriétés sur les agents, à savoir la facilité de développement, la vélocité et la suprématie.

La facilité de développement d'un agent est une mesure du rapport entre sa capacité à influencer son environnement et les autres agents, et la résistance potentielle qu'il peut rencontrer lors de ses actions sur d'autres agents. Plus formellement, la facilité F de développement d'un agent a est définie par :

$$F(a) = \sum messages_{renforants}(a) - \sum actions_{contraires}(a) - gravite(a)$$

où :

- $messages_{renforants}(a)$ sont les messages renforçants pour l'agent a ;
- $actions_{contraires}(a)$ sont les actions qui s'opposent au développement de l'agent a ;
- $gravite(a)$ représente en quelque sorte le fait que si un agent ne fait rien ou n'est pas considéré par d'autres agents, il perd de l'importance.

La vélocité de développement d'un agent est l'intégration de sa facilité dans le temps. Dans le cas la simulation proposée par F. Lesage, le pas de simulation était discrétisé, le terme infinitésimal valait 1. La vélocité, en théorie, est sensée donner des indications sur l'évolution à court terme sur les agents. Toutes ces indications pour l'ensemble des agents donnent donc une certaine prédiction sur la façon dont vont se comporter les aspects émergents de l'organisation. La définition de la vélocité V de développement en fonction du temps t est donnée par :

$$V(t+dt) = V(t) + F(t)$$

La suprématie est une représentation numérique à partir de considérations mécanistiques de la force qu'un agent aspectuel possède dans le paysage par rapport aux autres agents aspectuels. La définition de la suprématie S est :

$$S(t+dt) = S(t) + V(t)$$

Pour souligner l'aspect mécaniste de son approche, on peut faire le parallèle avec la mécanique classique :

- la facilité équivaut à l'accélération ;
- la vélocité correspond à la vitesse (intégration de l'accélération) ;
- la suprématie est analogue au chemin parcouru (intégration de la vitesse).

À partir de ces trois mesures définies – la facilité F, la vélocité V et la suprématie S – F. Lesage en proposait une quatrième, appelée « importance » qui

correspondait en quelque sorte à une combinaison linéaire des trois premières mesures pondérées par des coefficient $\alpha(t)$, $\beta(t)$ et $\gamma(t)$ (voir figure 2.2.13). Ces coefficients pouvaient varier dans le temps, permettant au système de prendre davantage en compte tel ou tel aspect du comportement des agents. L'importance I est définie par :

$$I(t) = \alpha(t)F(t) + \beta(t)V(t) + \gamma(t)S(t)$$

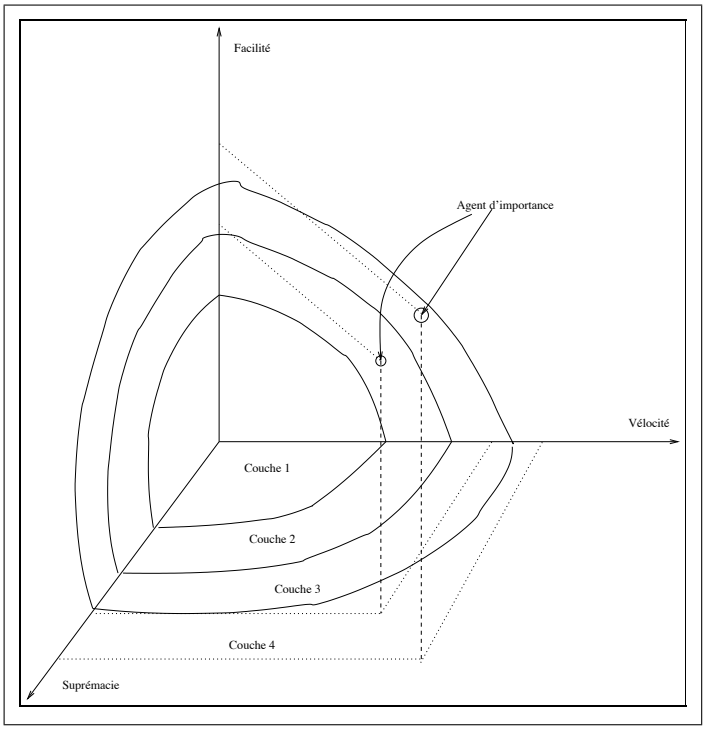


Figure 2.2.13 – *Schéma de l'importance (selon F. Lesage, image tirée de [Les00]).*

Ces mesures, qu'il appelle critères morphologiques – facilité, vélocité, suprématie et importance – sont fortement liés au fonctionnement des agents aspectuels.

Les agents aspectuels peuvent être agrégés en fonction de leur caractéristiques morphologiques. Ces groupes d'agents sont des points d'attention sur lesquels une analyse rationnelle sera effectuée. Ces groupes d'agents sont effectivement réifiés par des agents de la couche de morphologie.

La couche de morphologie comporte deux types d'agents : les agent pré-morphologiques et les agents morphologiques. On retrouve ces deux types d'agents pour chacun des espaces morphologiques correspondant aux quatre critères morphologiques définis (facilité, vélocité, suprématie et importance). Les agents pré-morphologiques correspondent à la projection des agents aspectuels dans l'espace morphologique, et les agents morphologiques réifient les agrégations des agents pré-morphologiques selon le critère correspondant.

La figure 2.2.14 illustre un exemple de déformation du paysage morphologique du point de vue d'un acteur du SIC.

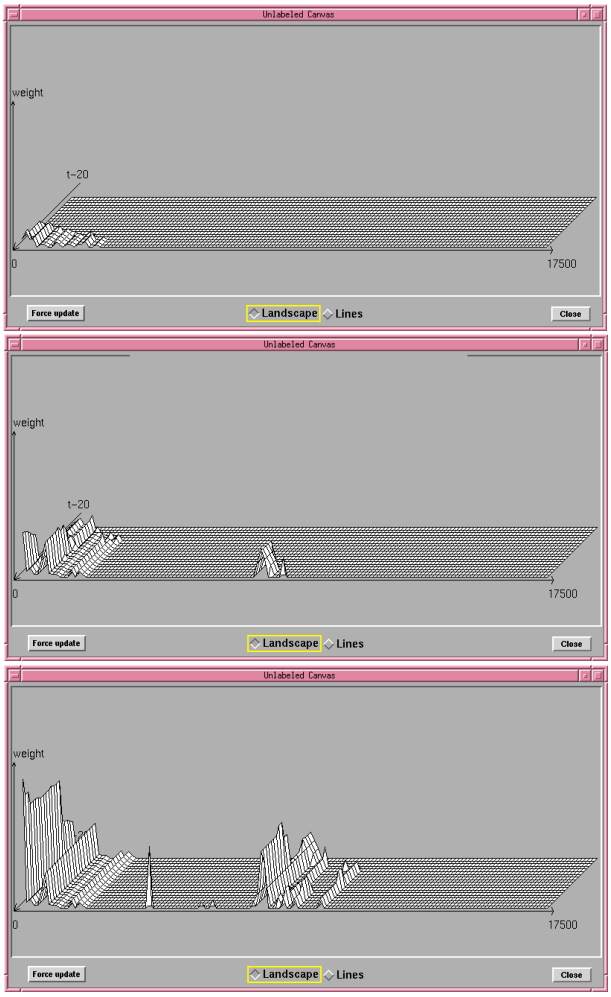


Figure 2.2.14 – *Séquence de déformation de paysage morphologique selon F. Lesage (images tirées de [Les00])* .

La couche d'analyse va procéder à la lecture de la description morphologique et dynamique de l'organisation aspectuelle. Les agents d'analyse se regroupent en quatre types :

- les agents d'analyse d'évaluation : ces agents produise une forme compréhensible des faits les plus marquants, mettant en évidence les contradictions, les concordances entre certains aspects de l'environnement ;
- les agents d'analyse de prédiction : ces agents tentent de prédire l'évolution du système en se fondant sur sa dynamique du système ;
- les agents d'analyse de validation : ces agents vérifient si les agents de prédiction ont évalué la situation correctement, et dans le cas contraire, ils peuvent prendre des mesures pour raffiner les prédictions futures ;
- les agents de décision : leur rôle est d'infléchir les deux couches précédentes afin de renforcer les croyances actuelles du système. Le système prend la décision de renforcer une lecture particulière de l'environnement en fonction d'une certaine correspondance entre ce qu'il perçoit et ce qu'il croit.

2.3 Remarques

Nous venons de voir quelques exemples d'applications de la morphologie (en biologie, en physique, en mathématique, en informatique...). L'étude de la forme est une méthode d'analyse relativement répandue en Science.

L'étude de la forme d'un système permet parfois de mettre en évidence certaines propriétés du système qui ne sont pas perceptibles à la première lecture. L'exemple le plus frappant est celui des systèmes chaotiques. Une première lecture pourrait faire penser que ces systèmes sont totalement erratiques, mais une deuxième lecture, géométrique, permet de mettre en évidence un certain ordre sous-jacent dans le comportement du système (voir l'exemple des attracteurs des générateurs de nombres aléatoires dans ce chapitre, section 2.2.4, par exemple).

La notion de forme d'un système permet de tenir compte du phénomène de stabilité structurelle, notion qui est essentielle dans l'observation et l'analyse des phénomènes. Un phénomène non structurellement stable ne peut être vraiment observé ni reconnu. Néanmoins cette notion sous-entend le choix d'un espace de représentation. Le choix de cet espace de représentation est important. Un mauvais choix peut en effet ne pas révéler de structures intéressantes.

Par sa généralité, l'étude de la forme, permet également d'étudier des systèmes qui peuvent être de natures très différentes et d'en observer les points communs. Ainsi, par exemple, il est tout à fait notable que le pendule et le système proie-prédateur présentent des analogies, comme on peut l'observer en comparant leur portrait de phase, et que l'étude de l'un peut apporter des connaissances supplémentaires sur l'autre. De plus, d'un point de vue expérimental, il se peut que certaines expériences soient plus faciles à mener sur un type de système, et que les observations obtenues soient transposables à d'autres systèmes de natures complètement différentes (au niveau du substrat).

Nous pourrions espérer de même, avec les systèmes multi-agents qui sont en fait de nature assez semblable aux systèmes étudiés en Physique, que l'observation et l'analyse d'une multitude de tels systèmes puisse conduire finalement à l'observation de motifs récurrents et communs, étayant ainsi une réelle théorie sur les systèmes multi-agents. Néanmoins, avant d'y parvenir, il faut disposer d'outils adéquats. Un des objet de cette thèse est de proposer, tout comme les travaux de Cardon, ou encore les thèses de S. Durand et de F. Lesage, une ébauche de développement de tels outils.

Deuxième partie

Proposition

Chapitre 3

Approche et modèle proposés

Dans ce chapitre nous présentons notre approche du contrôle des systèmes multi-agents de manière générale ainsi que le modèle proposé, en les détachant de toute application particulière. Des applications de ce modèle seront présentées par la suite dans les chapitres 4 (exemple réactif) et 6 (exemple fondé sur l'agentification aspectuelle).

Le terme « analyse » employé dans ce chapitre ne devra pas être confondu avec le terme « analyse » utilisé en génie logiciel. Sauf lorsque cela sera mentionné, nous emploierons le terme « analyse » dans le sens observation et étude d'un phénomène.

Nous commencerons par situer nos travaux (section 3.1), en précisant le type de système multi-agent où le genre de contrôle proposé s'applique le mieux. Ensuite nous proposerons de considérer les systèmes multi-agents auto-adaptatifs d'un point de vue morphologique (section 3.1.2) en introduisant différentes notions liées au comportement de tels systèmes. L'hypothèse centrale du contrôle morphologique sera introduite dans la section 3.2. Ensuite nous exposerons le modèle de contrôle à proprement dit (sections 3.3, 3.4, 3.5 et 3.6). Et finalement nous finirons sur des remarques générales sur d'éventuelles extensions du modèle (section 3.7).

Les applications de toutes ces notions sont présentées dans les chapitres suivants (chapitre 4 et chapitre 6).

3.1 Introduction

3.1.1 Cadre et buts

Ce que nous proposons est une approche relativement générale d'analyse et de contrôle de systèmes multi-agents considérés comme des systèmes complexes. Tout outil a son domaine d'application adéquat. Pour notre méthode, il s'agira de systèmes multi-agents de taille moyenne à des systèmes de très grande taille (plusieurs dizaines d'agents, centaines d'agents... voire plus). Le type d'agent constituant les systèmes étudiés sera entre le type d'agent réactif et l'agent très cognitif [Fer95]. Le contrôle recherché ne sera pas du contrôle optimal où le comportement de tous les agents est spécifié. Notre approche ne proposera pas de laisser non plus le système suivre son état comme il le peut. Nous pensons que la maîtrise de tels systèmes nécessite un contrôle qui se situe entre les deux aspects : il faut laisser les agents agir par eux même, mais il faut aussi corriger le système lorsque cela est nécessaire, et donc intervenir sur les agents. Et cela passe par une observation synthétique de la structure organisationnelle du système multi-agent.

Comme nous l'avons vu dans le chapitre 2, notre approche a des parentés dans d'autres branches de la Science, notamment en Physique. On ne peut s'empêcher de penser à la méthode de contrôle OGY des systèmes dynamiques par exemple (voir section 2.2.3). Cependant, la nature des systèmes contrôlés, bien que partageant des caractéristiques communes, est différente. En informatique on aura la chance de travailler sur des systèmes totalement artificiels. Notamment la brique de base des systèmes multi-agents, l'agent, pourra être totalement spécifiée ; à l'inverse de la physique, comme par exemple dans le cas des systèmes à particules, les particules constituant le système seront celles que le physicien trouvera dans la nature et aura pu enfermer dans sa « boîte » (pour reprendre la terminologie de Thom).

Étant donné un système multi-agent, nous nous posons la question de savoir ce qu'il se passe à l'intérieur, au niveau organisationnel, et éventuellement de construire un modèle à partir d'observations, modèle qui sera utilisable en vue de faire du contrôle non nécessairement optimal, c'est-à-dire par tendance. Il est à priori possible de savoir ce qui passe vraiment dans un système multi-agent. Comme il s'agit d'un système informatique que l'on a effectivement implanté, on connaît le fonctionnement du système à chaque pas de temps. Cependant mener un raisonnement sur les spécifications (ou le code source) est en général impossible

pour des systèmes complexes. L'idée est donc de générer un modèle à partir du système, par son observation. Notre approche est différente des autres approches, notamment des approches formelles, ou le modèle est construit non pas à partir d'observations mais des spécifications mêmes du système.

L'approche que nous adoptons est géométrique. Nous nous proposons d'étudier le système d'un point de vue morphologique. Cette approche a l'avantage d'être relativement générale. L'analyse que l'on propose cherche à corréler la forme de l'organisation d'agents avec son comportement. Le modèle que l'on construit permettra d'associer le comportement du système avec sa structure. Si l'on veut que le système se comporte de façon différente, on modifiera sa structure à l'aide du modèle.

3.1.2 Systèmes multi-agents auto-adaptatifs vus par la morphologie

Les systèmes auto-adaptatifs forment une classe particulière de systèmes complexes. Ces systèmes ont la propriété d'adopter un état stable de façon spontanée. On peut voir ces états comme des états d'équilibre pour le système. L'état adopté par le système auto-adaptatif dépend des conditions de l'environnement, qui imposent des contraintes sur l'état courant et éventuellement sur l'état initial du système. À noter que l'état d'équilibre n'est pas nécessairement un état statique : il peut s'agir d'un comportement dynamique mais stable. En fait, la description d'un système dynamique en temps que système auto-organisateur ou désorganisé [Ash62] est fondé sur la tautologie suivante : « tout système tend vers son état le plus probable ». Si cela nous arrange d'étiqueter cet état le plus probable comme un état organisé, nous dirons donc que le système en question est auto-organisateur. Néanmoins, il est incontestable que des états particuliers de certains systèmes présentent de fortes symétries, qu'il est difficile de nier et de qualifier ces état de désorganisés.

Nous proposons d'analyser le comportement de ce type de systèmes d'un point de vue morphologique. Nous allons représenter l'état d'un tel système dans un espace morphologique, ou espace des phases – nous avons vus des exemples de telles représentations dans la section 2.2. Les dimensions d'un tel espace sont arbitraires et leur choix dépend de ce que l'on veut représenter. Supposons que nous ayons choisi les dimensions de façon sensée, en fonction du système considéré. Considérons un système conçu pour résoudre un problème. Typiquement, le sys-

tème se trouve initialement dans un état donné (quelconque ou très particulier) qui ne produit pas la solution. D'un point de vue morphologique, en schématisant un peu, l'état initial et l'état solution seront respectivement des points particuliers de l'espace morphologique (voir figure 3.1.1).

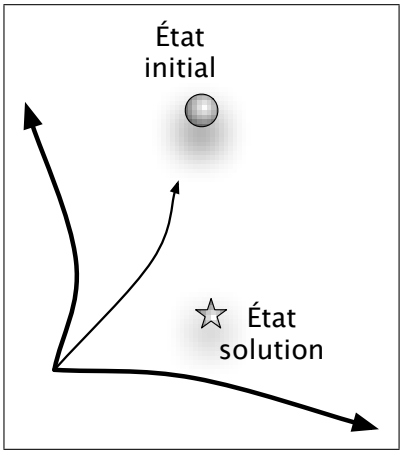


Figure 3.1.1 – *Représentation de l'état initial et de l'état final d'un système dans l'espace morphologique.*

Le système ne produisant pas la solution dans son état initial devra être modifié au cours du temps pour que son comportement et son état tendent vers la solution. Si le système résout le problème, on dira alors qu'il se trouve dans un état particulier que l'on appellera « état solution ». Il peut éventuellement y avoir plusieurs états solutions. Si nous avons une idée de la solution et une méthode de calcul pour transformer l'état du système en état solution, on fait tendre l'état du système plus ou moins vite vers cet état (voir figure 3.1.2).

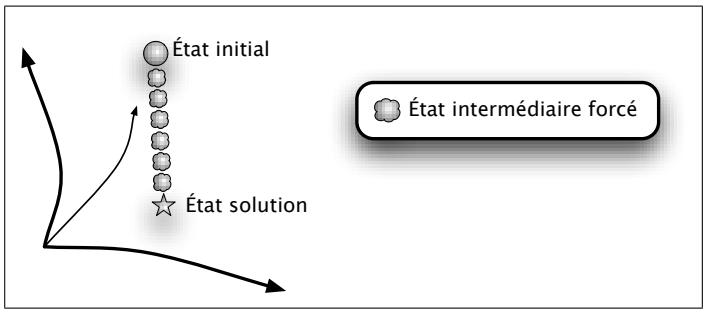


Figure 3.1.2 – *Succession d'états, imposés à partir de l'état initial pour arriver à l'état final.*

Dans le cas où il y a un ensemble d'états solutions, cela pourra se traduire par un ensemble de points solutions dans l'espace morphologique prenant la forme soit de point, de nuages de points, de formes connexes, etc... Cependant cela revient à peu près au même, cela dépendant des dimensions retenues pour constituer l'espace morphologique.

Dans le cas où l'état initial peut être variable, il est possible que l'état recherché (et donc en supposant que c'est l'état vers lequel le système converge), soit atteignable à partir de différents états initiaux. En reprenant la terminologie existante, ce point est qualifié d'attracteur, ou plus précisément de point fixe. L'ensemble des points de l'espace morphologique convergents est un bassin attracteur (voir figure 3.1.3).

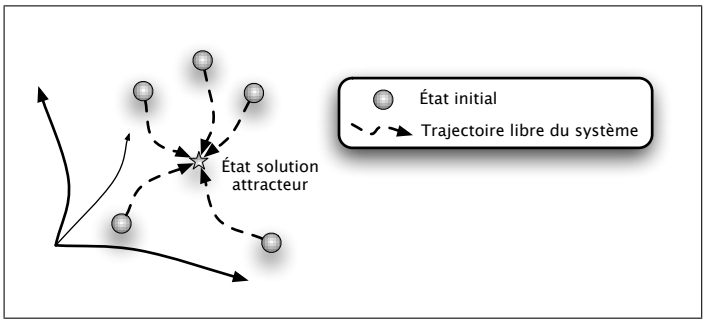


Figure 3.1.3 – *L'état recherche peut être atteint depuis différents états initiaux.*

Il peut y avoir évidemment plusieurs bassins attracteurs. Les bassins sont séparés par des crêtes qui délimitent leur zones d'attractions (voir figure 3.1.4).

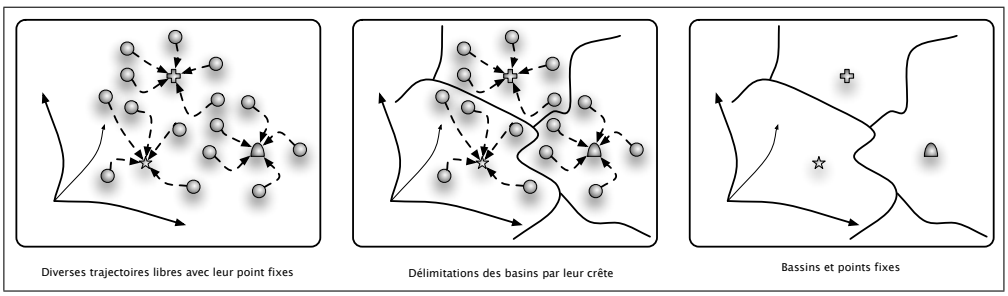


Figure 3.1.4 – *Plusieurs bassins attracteurs.*

Il se peut aussi, selon les dimensions choisies de l'espace morphologique, que l'état apparent stable du système soit en fait une succession périodique d'états stables. Cela se traduit par une boucle fermée dans l'espace des phases. On appelle cela un oscillateur. En général une fois que le système se trouve dans cette configuration, même si son état est modifié un peu, celui-ci retourne vers l'état stable. C'est typiquement le cas du pendule, par exemple (voir section 2.2.3).

3.1.3 Trajectoire libre d'un système

Considérons, à présent, un système auto-adaptatif en train de résoudre le problème pour lequel il a été conçu. Le concepteur de ce système a en fait intégré au système des mécanismes de modifications automatiques qui le font tendre de façon spontanée (car il est livré à lui même par définition) vers un état solution. Certains de ses paramètres changent, sa structure change au cours du temps. L'évolution de son état, le succession de ses états, peut être représentée par une trajectoire, ou une région, dans cet espace des états.

Du point de vue morphologique, construire un système auto-adaptatif, par exemple un système multi-agent auto-adaptatif, reviendrait à réaliser un système de sorte que son état emprunte une trajectoire dans l'espace morphologique qui le fasse arriver dans l'ensemble d'états solutions. Cette vision, qui pourrait sembler réductrice pour certains, ne cherche pas à dévaloriser les travaux de construction de tels systèmes, mais cela permet de prendre du recul sur ces démarches. Trouver une telle trajectoire n'est pas chose aisée. Nous pensons que beaucoup de travaux sur les systèmes multi-agents auto-adaptatifs tombent dans cette catégorie (produire des systèmes qui adoptent de façon spontanée une trajectoire arrivant dans l'état solution). D'autres travaux de recherches sur les approches orientées agents essayent de fournir des outils d'aide à la conception pour parvenir à cette tâche ardue (de façon indirecte, car elles n'utilisent pas ce point de vue morphologique).

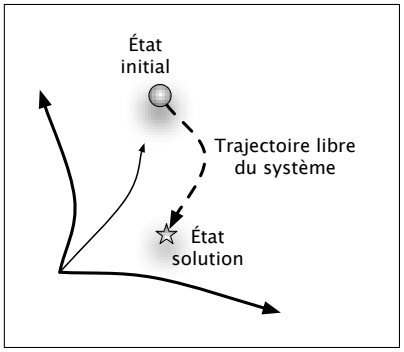


Figure 3.1.5 – *Schéma du comportement d'un système multi-agent livré à lui même. La trajectoire empruntée par le système est « spontanée ».*

Nous devons préciser la notion de trajectoire « libre » d'un système. Ce que nous entendons par le qualificatif « libre » est que le système n'est pas contraint par un autre système (un système qui examinerait son fonctionnement par exemple, hormis l'environnement sur lequel il n'y a pas de contrôle direct). C'est une trajectoire qu'il adopte de façon spontanée. Néanmoins cette spontanéité n'a rien de « magique », elle est tout à fait précisée (au moins implicitement) dans la définition du système. Cela suppose que l'on définisse précisément quel est le système, son environnement et ce qui ne fait pas partie du système, par exemple un second système observateur. Ce propos est schématisé sur la figure 3.1.6.

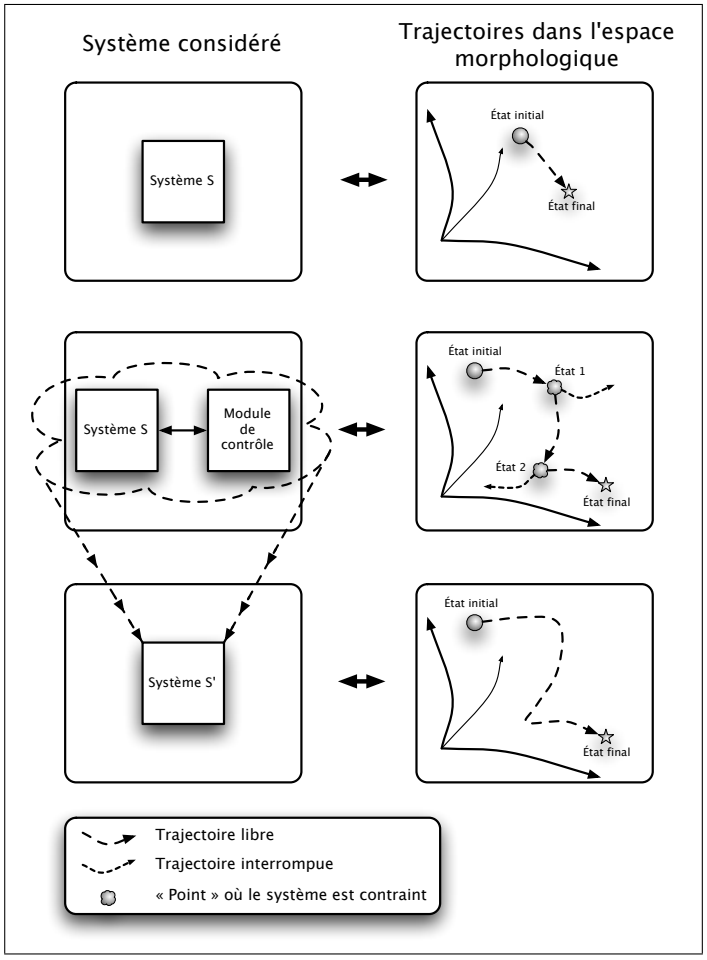


Figure 3.1.6 – *La notion de trajectoire libre dépend de la définition du système étudié et de ses limites.*

3.1.4 Modifications de trajectoires

Il se peut également que certains états initiaux ne permettent pas d'atteindre un état recherché. Par exemple le point visé est dans un autre bassin, ou ne peut pas du tout être atteint étant donné la structure du système, ou encore, est un point instable, répulsif (voir la figure 3.1.7).

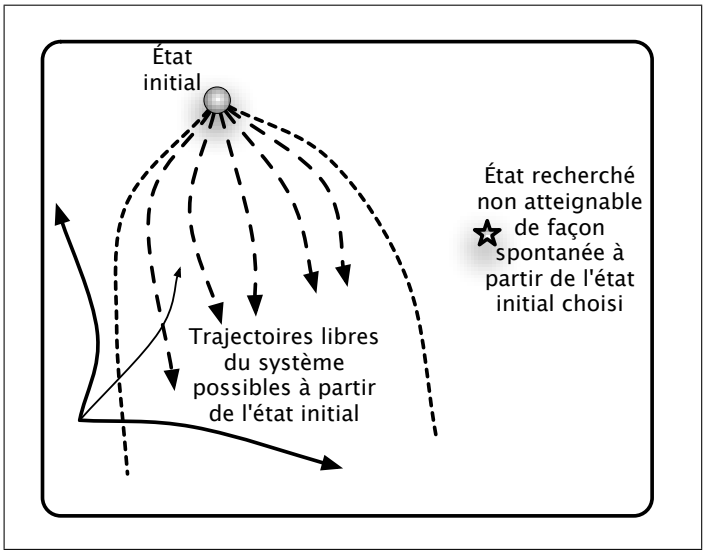


Figure 3.1.7 – *Parfois l'état recherché n'est pas atteignable depuis l'état initial. Le point recherché se situe à l'extérieur de l'enveloppe de toutes les trajectoires libres du système.*

Nous pouvons jouer sur la forme de la trajectoire en modifiant le système. Bien que la distinction ne soit pas franche, nous dirons que l'on peut jouer sur deux aspects du système : sa structure et son paramétrage (voir figure 3.1.8). Il faudrait, selon le cas et l'exemple étudié, préciser ces deux notions. Pour en donner une idée, nous pourrons dire, par exemple, qu'un paramétrage différent du système pourra impliquer un changement de valeur d'un seuil de comparaison dans l'automate comportemental des agents du systèmes ; et qu'une modification de structuration pourra consister en l'ajout d'agents particuliers ou un changement de hiérarchie entre agents.

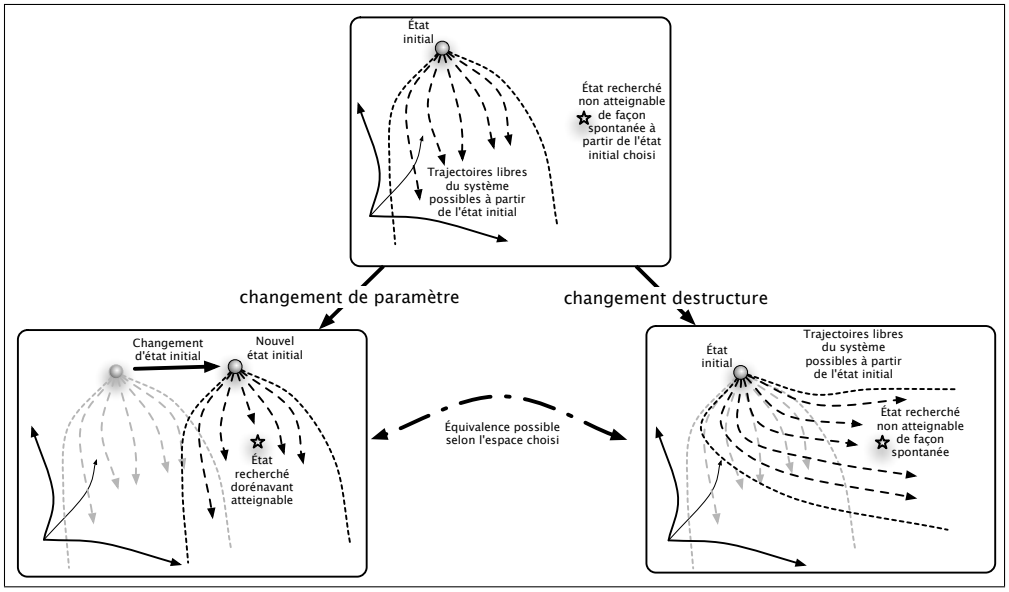


Figure 3.1.8 – *Comment remédier dans le cas où l'état initial ne permet pas d'atteindre l'état recherché. Soit on procède à un paramétrage différent, soit on modifie la structure du système multi-agent. La représentation morphologique des ces opérations est donnée à titre indicatif, car elle dépend des dimensions choisies de l'espace des états.*

3.2 Hypothèse fondamentale

Un système multi-agent peut-être considéré comme un certain processus complexe en cours d'évolution, qui peut être caractérisé, entre autre, par sa structure dynamique. Cette structure est déterminante dans le comportement émergent du système multi-agent. Notre modèle de contrôle se fonde justement sur cette idée. La forme d'une organisation d'agents (sa structure), engendre un comportement particulier. Un changement de comportement peut être engendré par un changement de structure. L'idée est donc de découvrir une corrélation possible entre la forme de l'organisation et de son comportement global. Si une telle corrélation est trouvée, l'étape suivante sera de l'exploiter en vue de faire du contrôle du système multi-agent (voir figure 3.2.1).

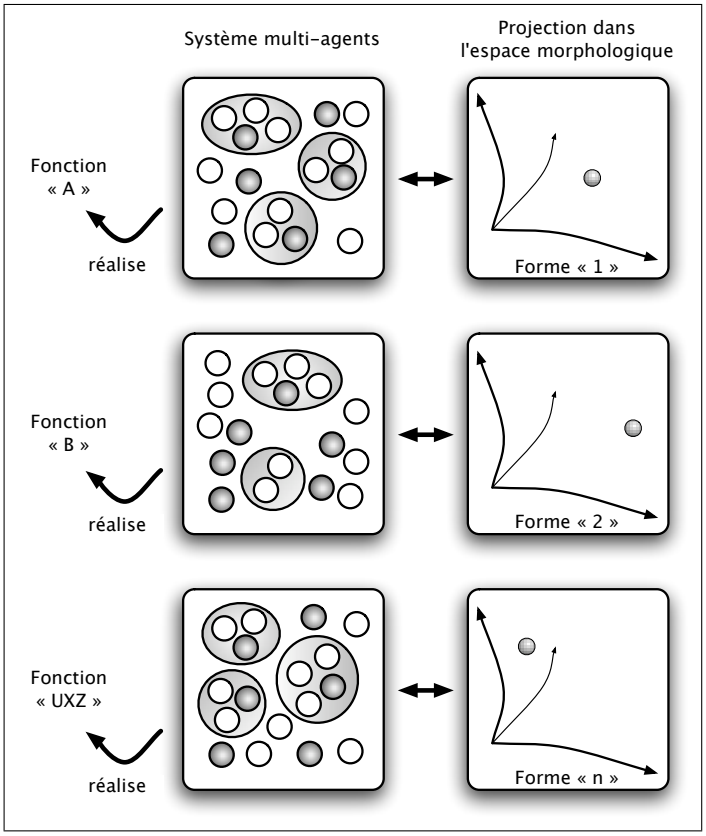


Figure 3.2.1 – *Mise en corrélation de la forme organisationnelle d'un système multi-agent et de son comportement global. Selon la structure de l'organisation, le comportement du système est différent. À chaque point de l'espace morphologique, spécifiant une structure de l'organisation, on associe la fonction réalisée par le système.*

Par ailleurs, comme nous l'avons mentionné dans l'introduction, nous pensons que la maîtrise de tels systèmes nécessite un contrôle qui se situe entre les deux aspects : il faut laisser les agents agir par eux même, mais il faut aussi corriger le système lorsque cela est nécessaire, et donc intervenir sur les agents. Et cela passe par une observation synthétique de la structure organisationnelle du système multi-agent. Sur le plan morphologique, cela se traduira par une suite de trajectoires libres entrecoupées d'états contraints à certains moments spécifiques (voir figure 3.2.2).

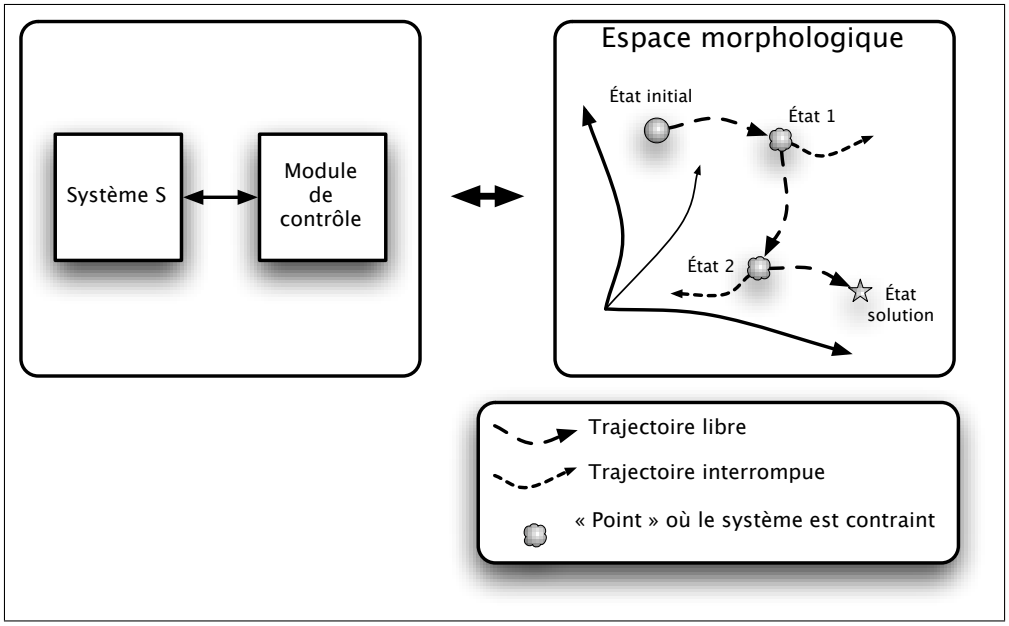


Figure 3.2.2 – *L'idée sous-jacente du modèle. On laisse le système évoluer librement. À certains moments le module de contrôle intervient pour changer la structure de l'organisation d'agents afin de changer son comportement et sa trajectoire dans l'espace morphologique.*

3.3 Organisation aspectuelle

L'organisation aspectuelle désigne le système multi-agent que l'on veut analyser et, par la suite, piloter. Cette appellation a son origine dans les travaux d'Alain Cardon [Car99].

Originellement le terme « aspectuelle » vient de la méthode d'agentification du problème étudié, réifié dans la dite organisation aspectuelle d'agents (ou encore appelé « organisation d'agents aspectuels »). Sans entrer dans les détails, nous dirons que dans cette méthode d'agentification, la correspondance, souvent rencontrée par exemple dans l'étude des problèmes de population, « un individu du modèle = un agent du système », ne tient plus. En revanche un élément du problème sera décomposé en plusieurs de ses aspects, et ce sont ces aspects qui seront agentifiés. L'entité initiale du problème sera représentée non plus pas un agent, mais par une multitude d'agents. Un trait particulier du système sera d'au-

tant plus significatif s'il est plus ou moins bien représenté par un groupe d'agents en activité. La définition de l'agent aspectuel, dans [Car03], est :

> Un agent aspectuel est un agent logiciel léger qui exprime, par son rôle, son action spécifique, ses communications et sa capacité agrégative, un caractère et un rôle fonctionnel et symbolique précis (dans un système générateur d'émotions et de pensées).

Alain Cardon propose dans [Car03] une méthode d'agentification pour construire une telle organisation. Dans le chapitre 5, nous en proposons une autre dans le même esprit que celle d'Alain Cardon, mais avec plus de précisions.

Néanmoins notre approche ne se limite pas uniquement à l'étude de telles organisations d'agents. D'autres types d'organisations peuvent être étudiées.

L'organisation aspectuelle est la réification du phénomène étudié par une méthode d'agentification aspectuelle. L'organisation peut être éventuellement couplée à un environnement (voir figure 3.3.1), par exemple dans le cas du pilotage d'un robot.

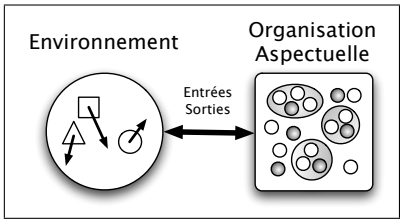


Figure 3.3.1 – *Schématisation d'une organisation aspectuelle couplée à un environnement externe. L'organisation peut interagir avec l'environnement via l'interface entrée/sortie.*

L'agent aspectuel, en plus de ce qu'il a à faire dans l'organisation, pour la simulation du phénomène étudié, doit pouvoir renseigner un observateur externe sur son état interne (soit en publiant régulièrement son état, soit en répondant à des requêtes, ou bien par le biais d'un dévermineur). Cette information est nécessaire afin de pouvoir construire une représentation de l'état de l'organisation aspectuelle.

3.4 Morphologie

3.4.1 Notion de projection

Idéalement, afin de faire l'analyse du système multi-agent, nous prendrions en compte tous les paramètres du système et des agents. Mais nous n'allons pas procéder comme cela, d'une part parce qu'il y a trop d'informations et certaines informations seront redondantes, et également il faut que l'information se présente sous une forme manipulable. Comme on est en informatique, il suffira d'observer le segment de mémoire où réside l'agent. Mais toutes les informations du segment de mémoire ne seront pas utilisées. On procède donc à une extraction d'informations, mathématiquement représentable par une projection d'un espace sur un autre souvent moins riche qualitativement en informations, mais plus pertinent pour le contrôle.

Alain Cardon avait donné dans [Car99] une liste générique de paramètres à prendre en compte dans l'analyse. Il l'avait augmentée dans [Car03] pour donner la liste suivante :

- apparence externe : sa situation d'acteur par rapport au autres agents et à l'environnement :
  - suprématie : mesure du fait que l'agent se situe ou non en position de force par rapport aux agents reconnus par ses accointances, qu'il a ou non de nombreux alliés et que ses ennemis sont ou non puissants ;
  - indépendance : mesure de l'autonomie de l'agent dans ses actions, précisant s'il lui est nécessaire ou non de trouver des alliés pour atteindre ses buts, si ses actions sont soumises par rapport aux actions d'autres agents, si sa structure de connaissance est incluse dans celles d'autres agents ;
  - persistance : mesure de la longévité de l'agent, de sa durée de vie qui peut être courte ou longue ;
- aspect interne : son état structurel par rapport aux fonctions qui lui sont assignées.
  - facilité : mesure du fait que l'agent a atteint son état courant en rencontrant plus ou moins de difficultés, de résistances.
  - vélocité : vitesse prise par l'agent pour atteindre les buts qu'il s'est lui même assignés ou qui lui ont été donnés ;
- état organisationnel interne : son fonctionnement propre entendu comme un processus d'échange d'informations entre ses modules qui forment sa

structure :
- intensité du flux d'activation interne : mesure de la quantité d'informations échangées entre ses composants internes lui permettant d'aboutir à une activation visible de l'extérieur par les groupes auxquels il appartient ;
- complexification : mesure de sa transformation structurelle engendrée par certains dysfonctionnements ou certaines situations difficiles ayant nécessité de transformer certains éléments de sa structure ;
- fréquence communicationnelle : mesure de la relation communicationnelle entre l'agent et les autres agents, en dehors de toute sémantique, représentant une mesure de son couplage dans l'organisation ;
- écart organisationnel : évaluation prenant en compte l'opérationnalité de la structure de l'agent et qui lui a permis de réaliser certaines actions sociales le conduisant à poser lui même une appréciation de la distance entre son propre état et l'état global de son environnement, tel qu'il se le représente.

Cependant, par manque de définition précise de la notion d'agent, ces caractéristiques ne sont pas « précisables » dans tous les cas. Nous retiendrons donc ces termes comme une indication pour la recherche de mesures pertinentes.

3.4.2 Propositions de type de formes

Nous nous intéressons à la forme que peut prendre une organisation d'agents. La notion de forme sera précisée jusqu'à être celle de forme géométrique précise.

Nous allons donner quelques exemples de mesures de formes concrètes et quelques types de formes que nous pouvons construire à partir de celles-ci (voir figure 3.4.1). Par exemple, à partir d'une valeur numérique propre de domaine $\mathbb{R}$ (taille mémoire occupée, valeur particulière d'une variable, durée de vie...), il est possible de construire des histogrammes. On peut étudier la distribution et donc la forme de ces histogrammes. En généralisant à plusieurs dimensions : l'espace est de domaine $\mathbb{R}^n$, où n est le nombre de dimensions (le nombre de mesures différentes effectués sur l'agent), on peut construire des nuages de points ou des histogrammes multidimensionnels. Si les agents portent une sémantique (par exemple déterminée par leur rôle), on peut étendre ce domaine par la dimension sémantique : $\mathbb{R}^n \times S$, où S est l'ensemble représentant la sémantique. Cela permettra de mettre en évidence, comment les aspects qui font sens au concepteur (les rôles des

agents, par exemple) sont en relation les uns des autres.

On peut également s'intéresser aux relations entre les agents : on dispose d'une valeur numérique par rapport à un autre agent. Il est alors possible de construire un graphe (préférence d'accointances, fréquence de communication...). À partir d'un tel graphe, on peut étudier d'autres propriétés de la structure, telles que la connexité, le diamètre...

Finalement, le développement, dans le temps, de la forme peut apporter des informations supplémentaires sur le comportement de l'organisation d'agents. Cela se rapproche de la notion de morphologie d'un phénomène se déroulant dans le temps, comme nous l'avions cité dans la section 2.1.

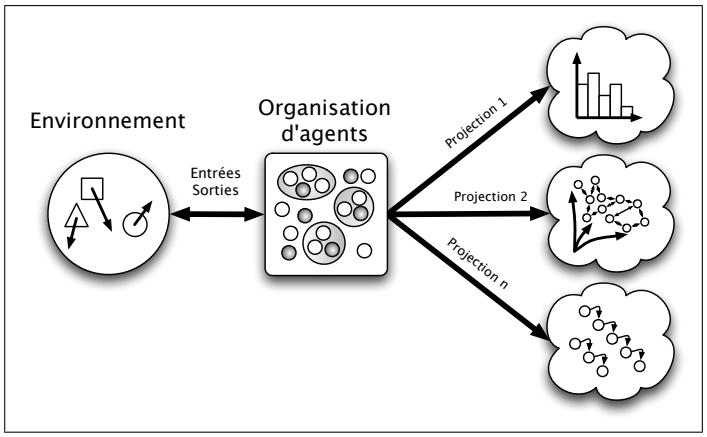


Figure 3.4.1 – *L'analyse morphologique d'une organisation d'agents passe par une projection d'un ensemble de mesures effectuées sur l'organisation. Différentes projections peuvent être employées selon la structure des agents et du type de contrôle que l'on envisage de faire. Les différents types de projections procurent différentes représentations de la forme de l'organisation (histogramme, graphe,...).*

Retour sur agents

Dans le cas où l'on envisage effectivement de faire du contrôle (et non juste de l'observation), la mesure effectuée sur l'agent qui sert à la construction de la forme devrait correspondre à une mesure qui peut varier de façon contrôlée, directement ou indirectement, en modifiant un certain paramètre de l'agent ou bien de son

environnement (voir figure 3.4.2). Par exemple, ou bien la mesure porte sur un paramètre de l'agent modifiable directement et le contrôle et la corrélation s'en trouvent simplifiés ; ou bien la valeur mesurée n'est pas directement modifiable, et il faut peut-être intervenir sur d'autres agents, ou bien sur l'environnement de l'agent, ou bien encore supprimer ou rajouter des agents particuliers pour modifier les valeurs afin de faire tendre la structure globale de l'organisation vers la forme recherchée.

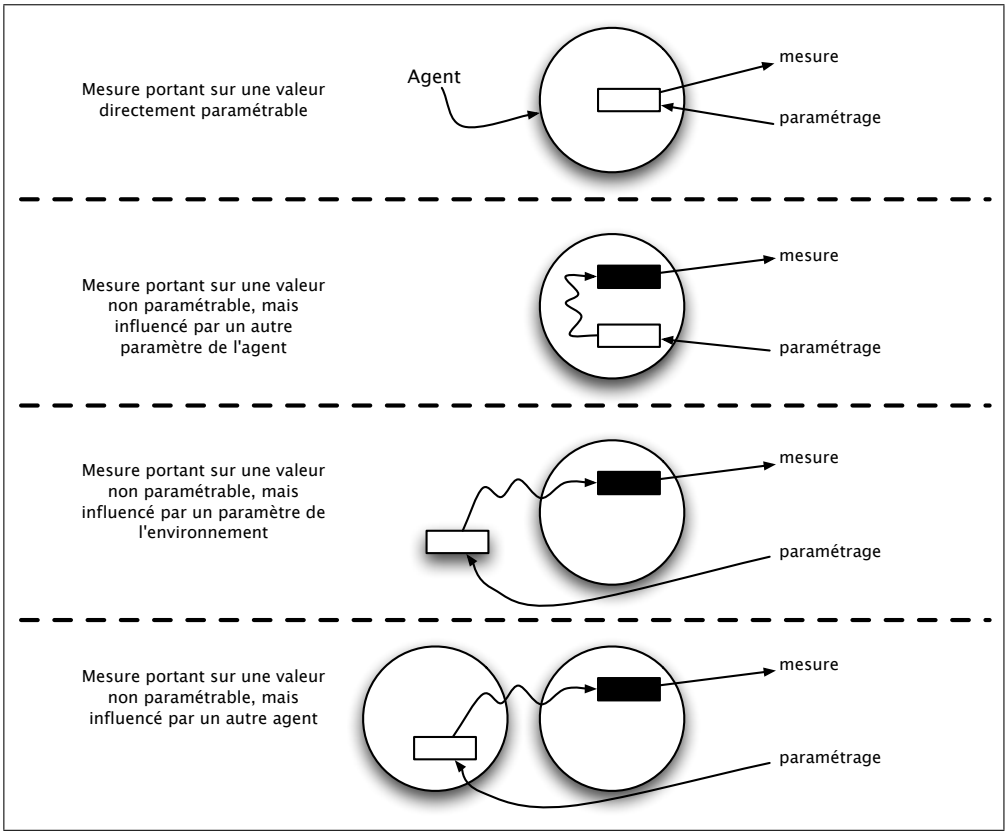


Figure 3.4.2 – *Différentes possibilité de modification d'une mesure (directement ou indirectement).*

Métrique, équivalence

Afin de pouvoir manipuler les formes qui décrivent la structure et la dynamique de l'organisation, il est souhaitable que l'espace de ces formes soit doté d'une métrique, ou du moins d'une topologie. Cette métrique sera utile pour faire

de la classification, de l'apprentissage, contrôler la convergence du comportement. Si la forme est construite à partir d'un nuage de points, on pourra par exemple s'appuyer sur des méthodes statistiques (ou mécaniques) pour faire une comparaison [MH87]. Si le type de forme est fondé sur des graphes, il existe des algorithmes de comparaison de graphes, mais les calculs sont plus ardus. En effet, la comparaison de graphe [MB00, DJD+un, CH98, CKP95, PSZ99, Ben02, WH97, SCC97], ou le calcul d'isomorphisme de graphes ou de sous-graphes, sont des problèmes difficiles et gourmands en ressources.

À partir de la métrique, on peut aisément définir une classe d'équivalence, en posant comme définition d'équivalence : deux formes A et B sont ε-équivalentes si $d(A,B) < \varepsilon$.

Sinon, s'il s'avère très difficile d'avoir une pseudo-métrique, on pourrait se contenter de la définition d'une relation d'équivalence entre deux formes. Cela donnera un moyen minimal pour classer les formes.

3.5 Analyse du système

3.5.1 Apprentissage, construction du modèle

L'observation du système permet une construction expérimentale du modèle. Alors que le système fonctionne, sa structure organisationnelle est observée et analysée. Les formes observées, qui décrivent la structure du système, sont mises en correspondance avec le comportement global observé du système. Cette corrélation peut être grossière ou fine. Un moyen de réaliser cette corrélation serait de catégoriser les formes selon que le système effectue bien sa tâche ou non. Une autre manière moins dichotomique de classer les formes serait de proposer différents niveaux d'appréciation, avec une pondération quantifiable si possible.

3.5.2 Cinématique et dynamique

En reprenant la terminologie de la Physique, nous pouvons dire que la succession de formes extraites donne expérimentalement la cinématique des formes

du système. La cinématique est l'ensemble des formes possibles du système. La dynamique est l'étude du passage d'une forme du système à une autre forme.

Le modèle cinématique et dynamique dépendra en fait de la définition que l'on donnera à la notion de forme que l'on cherche à observer. On peut tout aussi bien considérer toutes les formes possibles, ou juste les formes qui sont statistiquement représentatives.

Cinématique

La construction expérimentale du modèle cinématique est relativement simple et directe. Il suffit, de façon assez basique, d'observer le système (voir figure 3.5.1).

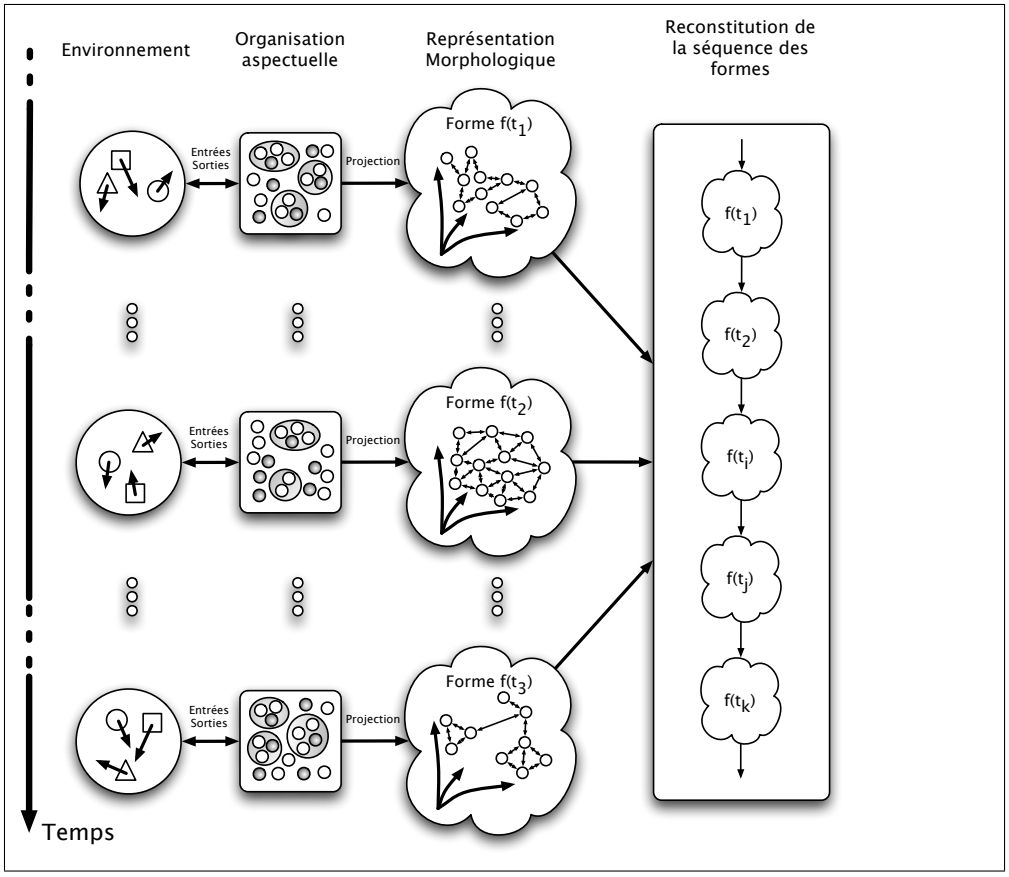


Figure 3.5.1 – *Au cours du temps, le système multi-agent évolue. De façon régulière, on construit la forme représentative de l'état de l'organisation. En tenant compte des transitions entre les formes, on construit le modèle dynamique du système.*

Il est clair cependant que la construction des modèles fondée sur l'observation ne pourra garantir que le système agira de telle ou telle manière de façon certaine, contrairement à ce que les approches formelles tentent de faire. Le modèle sera en fait en perpétuelle révision. De nouvelles observations pourront modifier le modèle de façon plus ou moins significatives. Néanmoins si les phénomènes sont observés de façon récurrente, il n'est pas irréaliste de penser que le modèle ainsi construit sera relativement représentatif du comportement du système. La contrepartie de cette « incertitude » est de fournir une méthode opérationnelle, contrairement aux méthodes formelles qui buttent contre la complexité en nombre de calculs.

Dynamique

En analysant la transition d'une forme à une autre, nous pouvons déterminer la dynamique des formes du système (voir figure 3.5.2). L'étude de la dynamique permet de révéler le comportement du système à court terme et éventuellement à moyen terme et, selon les cas, à long terme (par exemple dans le cas où le système présente un unique attracteur). Le modèle dynamique peut être sous deux formes, une simple et une plus sophistiquée.

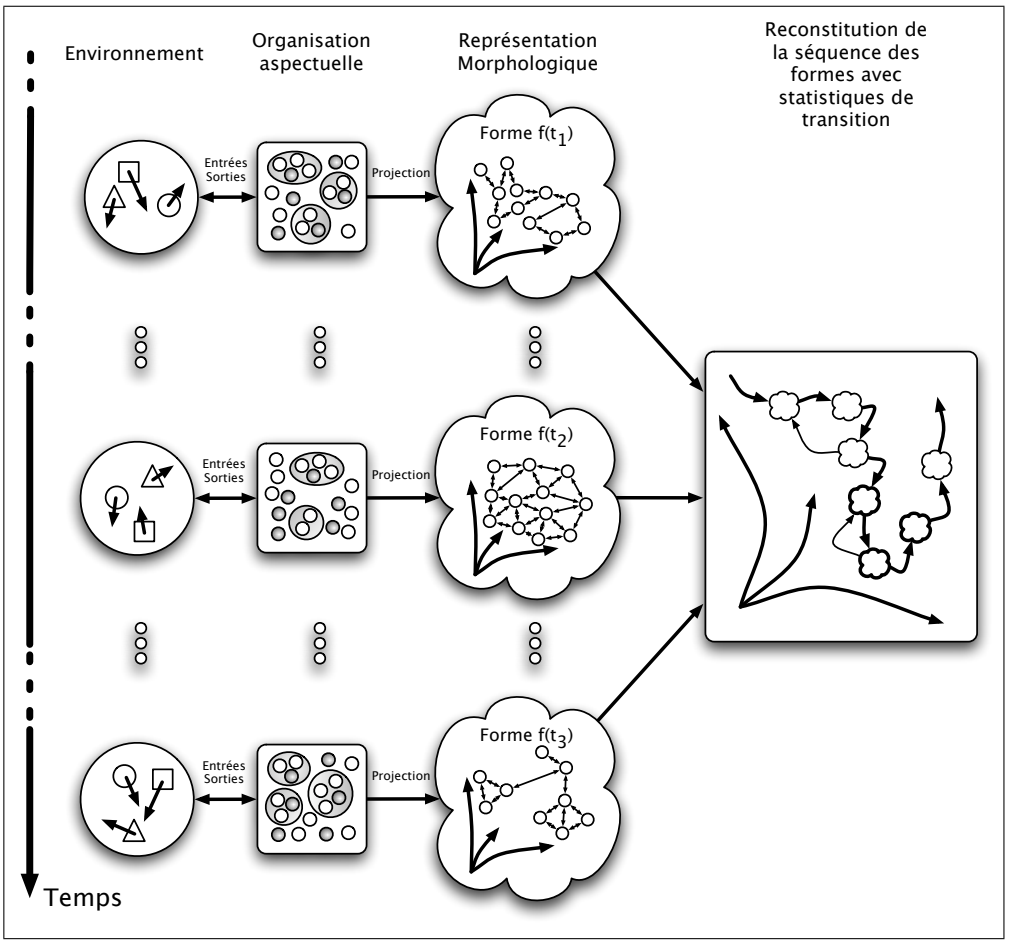


Figure 3.5.2 – *Au cours du temps, le SMA évolue. De façon régulière, on construit la forme représentative de l'état de l'organisation. L'ensemble des ces formes constitue la cinématique du système.*

La forme simple est un modèle probabiliste qui donne les probabilités de transition d'une forme à l'autre. La construction de ce modèle est relativement simple. Il est établi à partir des statistiques sur les formes, en faisant fonctionner le système suffisamment longtemps.

L'autre forme du modèle dynamique, plus difficile à établir, est plus explicative. Ce modèle s'intéresse aux causes de transition d'une forme à une autre. Si le type de forme décrivant la structure organisationnelle du système n'est pas trop pauvre (c'est à dire qu'elle contient suffisamment d'informations), il serait envisageable de corréler certaines modifications d'une forme avec d'autres modifications d'une autre forme possible issue de la transition. L'intérêt de ce modèle est double. D'une part il fournirait davantage d'explications sur la transition, et d'autre part il fournirait un moyen de provoquer une transition particulière pour aboutir à une forme voulue du système. Pour forcer le système à prendre une forme, on peut envisager de modifier toute la structure de façon forcée ; mais un moyen plus économique et plus intéressant serait de modifier ce qui est strictement nécessaire pour ensuite que le reste de la transformation se fasse de façon automatique, minimisant ainsi l'intervention extérieure.

3.5.3 Graphe d'évolution : le modèle

Dans les cas simples où il n'y a que quelques paramètres à prendre en compte, on peut représenter l'évolution du système dans l'espace des phases des paramètres. Et dans les cas à faible dimension, on peut même tracer la trajectoire du système dans un espace à 1, 2 ou 3 dimensions. Mais dans les cas de plus grande dimension (grand nombre d'agents, grand nombre de mesures), de telles pratiques ne sont plus possibles. On peut toujours procéder à une réduction de dimensions par des méthodes telles que l'analyse en composante principale [MH87]. Mais il se peut que ce type de projection ne soit pas adapté. Nous avions tenté d'appliquer l'analyse en composante principale, mais lorsque le nombre de rôles des agents augmente, il arrive souvent qu'on ne parviennent pas à extraire des plans principaux. Une façon de s'en sortir est de ne considérer qu'une partie de l'organisation d'agents. Par exemple on peut s'intéresser uniquement à certains agents dont le rôle est de tel ou tel type, et procéder à l'étude des modèles construits à partir de ces sous-ensemble.

Une autre approche, qui permet également de contourner le problème de haute dimension, est de ne plus considérer l'évolution du système dans l'espace des pa-

ramètres mais dans l'espace des formes. On peut représenter l'espace des formes simplement et naturellement par un graphe : les noeuds du graphes sont les classes d'équivalences des formes et les arcs sont les transitions possibles d'une forme à une autre. Les arcs peuvent êtres libellés par la fréquence (probabilité) de transition entre deux formes. En appliquant l'hypothèse centrale de notre modèle, nous pouvons attacher à ces nœuds, à ces formes, des étiquettes qui correspondent au comportement global du système (voir figure 3.5.3). Selon les dimensions choisies pour décrire la forme, il se peut que l'espace des formes n'ait pas de limite de dimension possible. Mais le système peut ne pas passer par toutes les conformations possibles, ce qui limiterait la taille de l'espace des formes. Cependant la représentation sous forme de graphes d'évolution perd un peu d'information géométrique. Nous pouvons dire que le modèle sous forme de graphe est un modèle topologique du système.

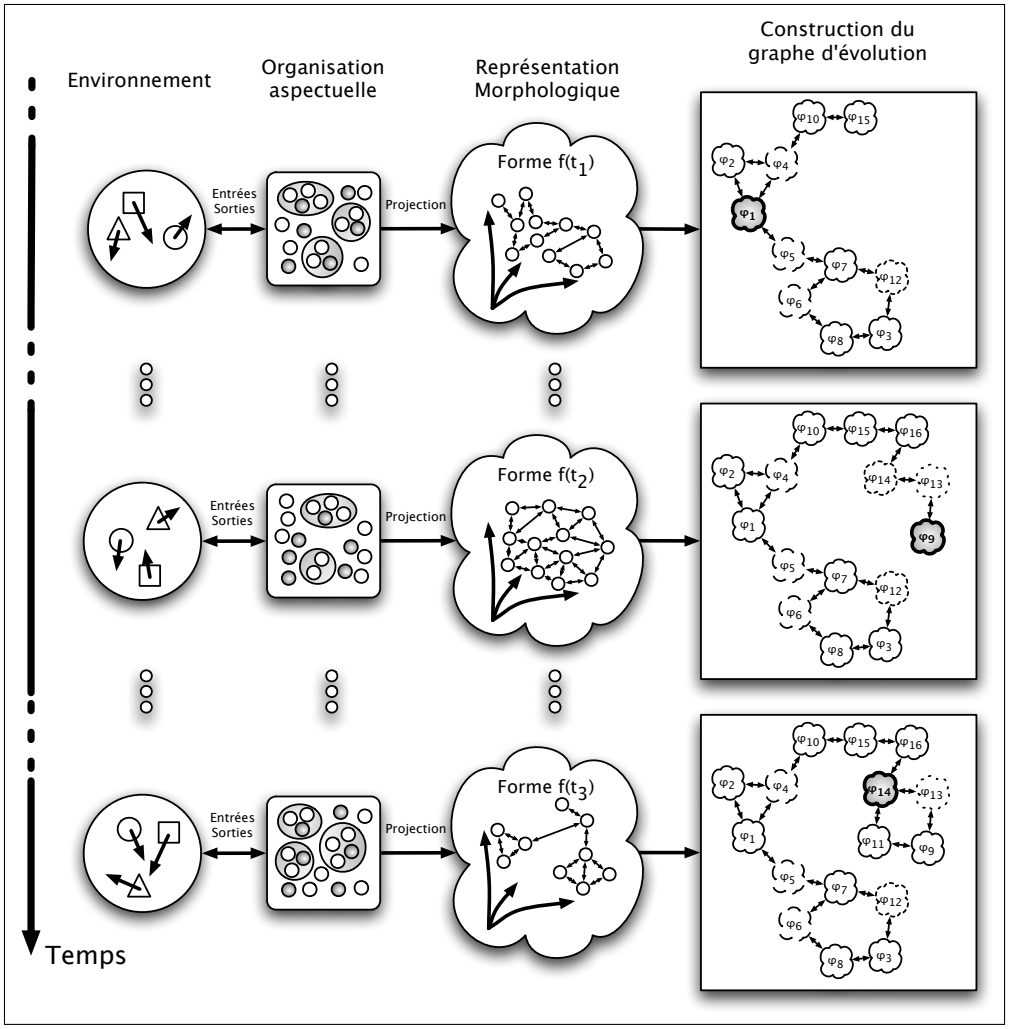


Figure 3.5.3 – *Construction du graphe d'évolution. Au cours du temps, le système multi-agent évolue. À chaque instant la forme représentative de l'organisation est construite. S'il s'agit d'une nouvelle forme, celle-ci est ajoutée au modèle. Les attributs des formes liés à la sémantique du problème ainsi que ceux des transitions entre les formes sont mis à jour.*

La corrélation de la structure de l'organisation est explicitée dans le modèle en attribuant des étiquettes aux nœuds du graphe d'évolution, qui décrivent le comportement global du système (voir figure 3.5.4).

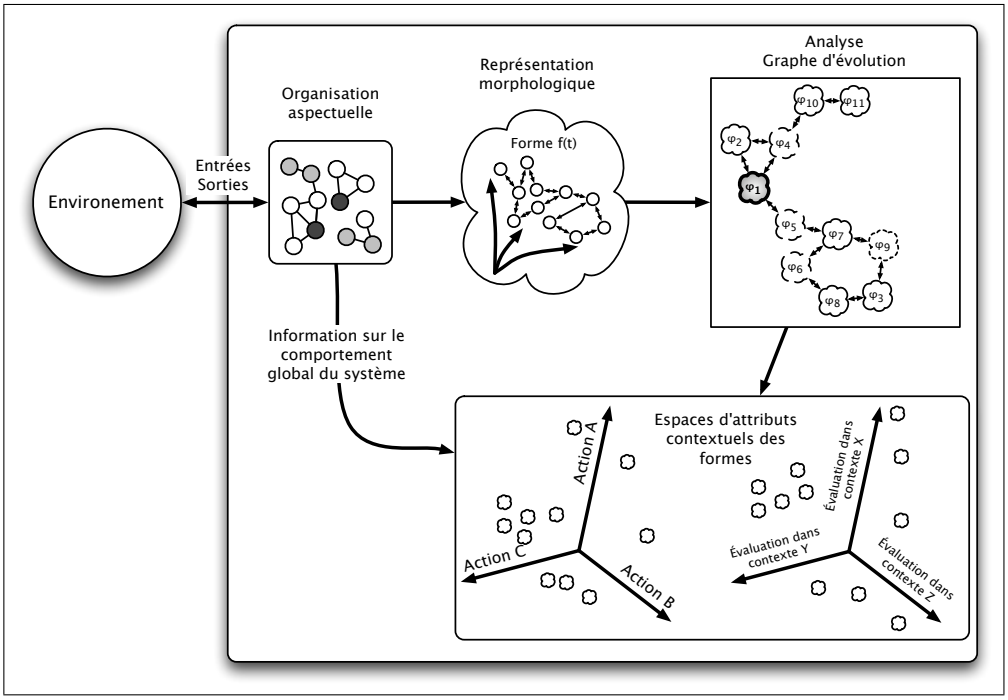


Figure 3.5.4 – *Graphe d'évolution. On explicite la corrélation entre la forme de l'organisation et son comportement en attribuant aux nœuds du graphe des attributs spécifiant le comportement global du système.*

3.6 Contrôle du système

Dans cette section nous allons voir comment procéder pour réaliser une boucle de contrôle de systèmes multi-agents à partir de la description morphologique. Une fois que le modèle (le graphe d'évolution) est extrait, il est possible, dans une certaine mesure, de contrôler le système.

3.6.1 Hypothèse du contrôle

La boucle de contrôle que nous construisons est fondée sur l'hypothèse déjà mentionnée, mais que nous rappelons ici : le comportement du système multi-agent est corrélé à sa structure. Dans une certaine mesure, une modification de sa

structure engendre une modification de son comportement. Le contrôle du système passera donc essentiellement par une modification de la structure de l'organisation. Il serait possible de combiner ce type de contrôle avec d'autres genres plus radicaux comme l'ajout ou le retrait d'agents, ou l'envoi de messages spécifiques à certains agents, sans passer pas la morphologie.

3.6.2 Mécanismes de contrôle

Jusqu'à présent, le système est constitué des modules suivants :

- l'organisation aspectuelle, qui est le système multi-agent destiné soit à représenter un phénomène, soit à piloter un système tel un robot ;
- un module de représentation, le module de morphologie, qui décrit la structure de l'organisation aspectuelle et sa dynamique.

Un troisième module est nécessaire pour faire le contrôle, c'est le module d'analyse. Ce module va prendre en compte la description morphologique pour faire le pilotage de l'organisation aspectuelle (voir figure 3.6.1).

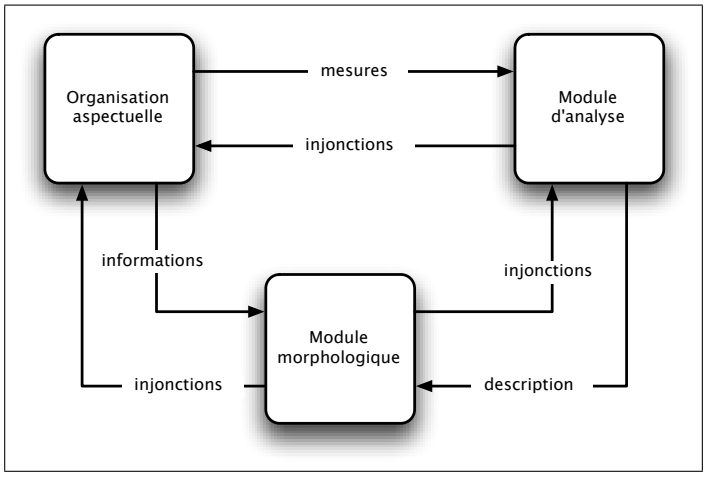


Figure 3.6.1 – *L'organisation aspectuelle représente le phénomène simulé. Le module morphologique produit une description géométrique de l'organisation aspectuelle. Un troisième module, le module d'analyse, est inséré pour réaliser le contrôle de l'organisation aspectuelle à l'aide du module de morphologie.*

La bonne adéquation du comportement du système par rapport à ce qu'il doit faire est déterminée par l'indication donnée par certaines règles définies pas le concepteur du système. Ces règles sont vraiment spécifiques au problème pour lequel le système multi-agent a été conçu. L'analyse interprète ce que l'organisation aspectuelle fait. On retrouve la sémantique à ce niveau.

Si le système opère correctement, il n'est pas nécessaire de changer son comportement. Il n'y aura pas de retour sur le système, on laisse celui-ci continuer librement à évoluer. De part la nature « autonome » des agents constituant le système, ce dernier évoluera selon ses tendances.

En revanche si le système ne se comporte pas comme il le faudrait, le module de contrôle entre en jeu. Si le système ne se comporte pas correctement, la forme du système décrite par le module de morphologie est considérée comme « mauvaise » par le module d'analyse. L'analyse opère un choix sur la forme que le système devrait avoir. Si le modèle du système, construit expérimentalement, n'est pas trop « mauvais », l'analyse choisit une forme adéquate en fonction des attributs de la forme. Par exemple, l'analyse pourra choisir une forme proche de la forme actuelle mais qui est considérée comme adéquate pour la situation donnée. Le choix d'une forme proche de la forme actuelle permettrait de limiter l'intervention de l'analyse sur le système. L'idéal serait une intervention chirurgicale, bien ciblée, sur quelques agents clefs. Cette idée est en accord avec celui du contrôle non linéaire des systèmes complexes que l'on peut rencontrer en physique. Un petit effet sur le système peut avoir de grandes conséquences sur le comportement du système à moyen et long terme (voir section 2.2.3).

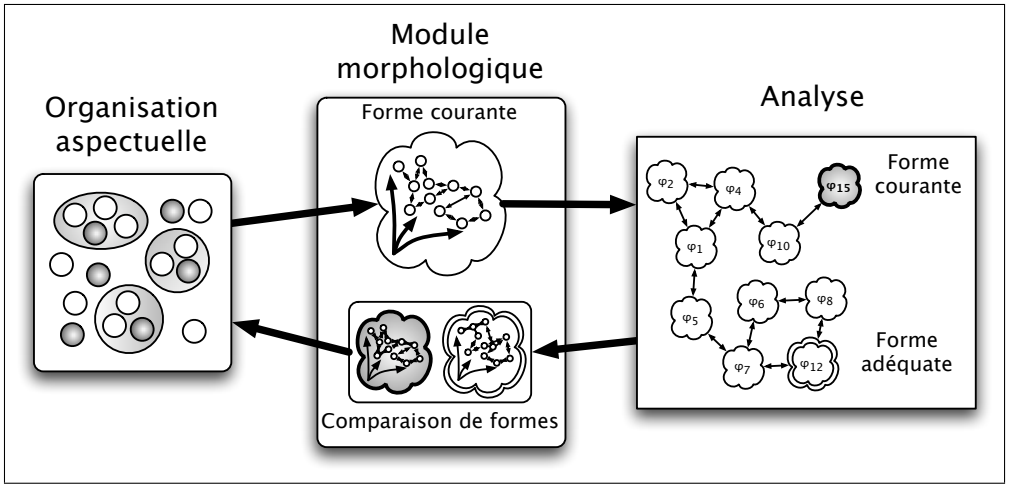


Figure 3.6.2 – *Schéma du modèle de contrôle.*

3.7 Extensions

Cette section présente quelques pistes d'extension du modèle sous sa forme élémentaire.

3.7.1 Agentification de la morphologie et de l'analyse

Dans [Car03] Alain Cardon propose de réifier le module de morphologie et d'analyse par une organisation d'agents, comme l'a fait par exemple F. Lesage dans sa thèse avec les agents morphologiques et de pré-morphologies [Les00]. Dans le cas simple où cette fonctionnalité est remplie par un seul agent, nous n'avons pas vu l'intérêt de passer par un « agent », si ce n'est pour donner une certaine homogénéité au système. Une idée exploitant davantage le paradigme multi-agent serait de remplacer le module de morphologie par une organisation d'agents du type aspectuel, mais qui opère sur les traits morphologiques de l'organisation aspectuelle, et de même pour le module d'analyse.

3.7.2 Sondage morphologique

Il se peut que la mise en place d'un module de morphologie et d'analyse se révèle trop lourde dans la pratique, notamment dans le cas de systèmes multi-agents avec des restrictions particulières comme l'isolement, la couverture géographique... Si la mise en place d'un module de morphologie se révèle trop lourde (par exemple, trop de mesures à faire) il est possible de mettre en œuvre un système d'échantillonage, et, par extension, faire une extrapolation sur le système entier.

Chapitre 4

Contrôle morphologique d'un système multi-agents réactif

Dans cette partie nous présentons un premier exemple d'application du modèle dans le cadre du contrôle d'une population d'agents relativement simples. « Historiquement », c'est sur cet exemple que nous avons validé certaines de nos idées. Cet exemple nous a permis de spécifier certains points du modèle. Nous présentons le cadre du problème (section 4.1), les caractéristiques du système multi-agents (section 4.2). Ensuite nous décrivons complètement la méthode d'analyse et de contrôle développée pour cet exemple (section 4.3) et nous présentons quelques résultats expérimentaux (section 4.4).

4.1 But de l'exemple et description générale

Nous allons présenter un exemple de contrôle d'une organisation d'agents à l'aide de la morphologie.

L'exemple porte sur le contrôle du niveau d'une population d'agents. La population peut varier en taille. Le nombre d'agents présents dans l'organisation varie au cours du temps du fait de leurs interactions au sein de l'organisation. Pour reprendre le schéma du modèle (chapitre 3), l'organisation à contrôler, dont le niveau de la population fluctue, est l'organisation aspectuelle. Cette organisation

sera analysée par un module morphologique et pilotée par un module d'analyse. Dans cet exemple simple, les modules de morphologie et d'analyse ne seront pas réifiés par des agents ou des organisations d'agents.

L'exemple, volontairement simple, illustre concrètement comment il est possible d'analyser et de contrôler une organisation d'agents en construisant une représentation géométrique de l'état de l'organisation et en travaillant sur ses formes afin de contrôler l'organisation. Le terme de pilotage peut être plus approprié, dans ce cas, que le terme contrôle. Comme nous allons le voir, l'organisation n'est pas exactement contrôlée de manière absolue car les agents sont relativement réactifs et on leur laisse une part « d'autonomie ». Cela rend l'organisation très volatile et difficilement maîtrisable. Cependant, dans une certaine mesure, le pilotage est possible.

4.2 Organisation d'agents

4.2.1 Structure des agents

Les agents utilisés dans cet exemple, sont simples et réactifs. Un agent possède un niveau d'énergie et peut communiquer avec les autres agents pour initier des échanges ou bien « se battre ». Le comportement d'un agent possède un degré de liberté qui permet de moduler ses choix.

Au niveau de l'implantation, un agent est une structure de données qui est mise à jour à chaque cycle de simulation par un automate. La structure de données comporte, entre autre, le niveau d'énergie de l'agent, son paramétrage, les références des agents avec lesquels il entreprend des échanges...

4.2.2 Environnement

Les agents sont situés dans un environnement commun où ils peuvent interagir les uns avec les autres (voir figure 4.2.1).

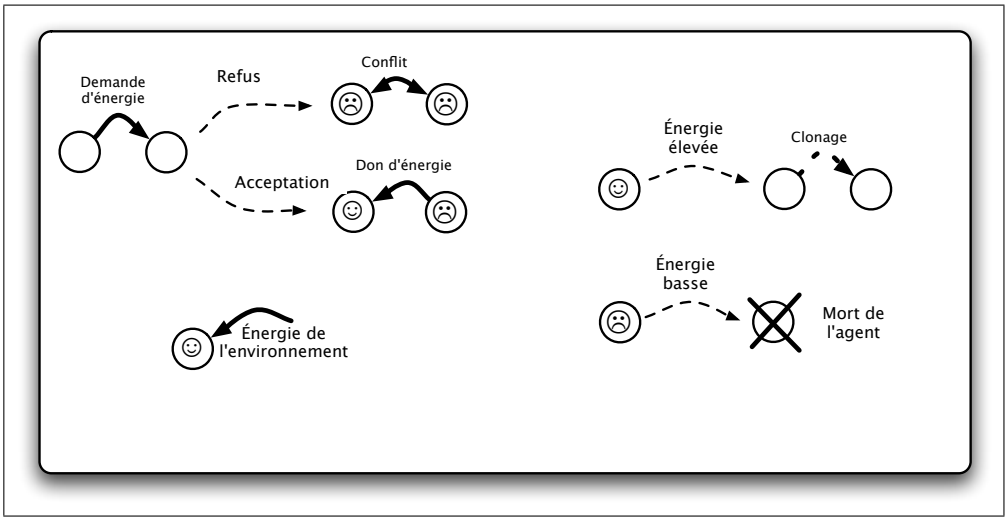


Figure 4.2.1 – *Ensemble des interactions et actions possibles dans le système multi-agents (demande, don, gain, perte d'énergie ; conflit, clonage et mort).*

Les interactions et actions possibles dans ce système sont :

- un agent demande de l'énergie à un autre agent ;
- un agent peut donner de l'énergie à l'agent qui lui en a demandé ;
- un agent peut refuser la demande, et dans ce cas les agents se battent et perdent tous deux de l'énergie ;
- un agent récupère de l'énergie de l'environnement ;
- un agent peut se cloner, s'il a assez d'énergie ;
- un agent qui n'a plus d'énergie meurt et il est retiré de l'organisation.

4.2.3 Énergie

Les agents, dans l'environnement commun, partagent une ressource, appelée de façon imagée « énergie ». À chaque cycle, une quantité fixe d'énergie est mise à disposition des agents. Cette énergie est distribuée de façon uniforme parmi les agents.

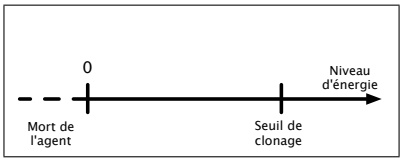


Figure 4.2.2 – *Niveau d'énergie. Si le niveau d'énergie est négatif, l'agent meurt. Si le niveau d'énergie dépasse le seuil de clonage, l'agent a la possibilité de se cloner.*

4.2.4 Création initiale des agents

Lorsque les agents sont initialement créés, leur niveau d'énergie est non nul. Les paramètres de leur comportement sont initialisés de façon aléatoire sur un intervalle approprié. Typiquement, le nombre d'agents initial est de 50 à plus (voir résultats plus bas).

4.2.5 Mort d'un agent

Un agent meurt lorsque son niveau d'énergie devient nul. Un agent mort est retiré de l'organisation (voir figure 4.2.2). La taille de la population diminue donc d'une unité. Lorsque toute la population d'agents a disparue, plus aucun agent ne peut être créé.

4.2.6 Clonage

Un agent a également la possibilité de se « cloner » lorsqu'il a suffisamment d'énergie (voir figure 4.2.2). Le clone de l'agent est identique à l'agent original (au niveau des ses paramètres), hormis son niveau d'énergie qui peut être différent. À la suite du clonage d'un agent, la taille de la population augmente de un. Un agent ne se clone pas forcément s'il a assez d'énergie, il y a une certaine probabilité pour que cela se réalise (typiquement une probabilité de $0,5$).

4.2.7 Échange d'énergie

Les agents peuvent également interagir entre entre eux pour des échanges d'énergie. Le comportement d'un agent est paramétré par une variable appelée « ardeur ». Plus un agent est ardent, plus il aura tendance à demander à un autre agent de l'énergie. Inversement, moins son ardeur est grande, moins il aura tendance à initier une demande. Un agent A peut demander à un autre agent B de l'énergie. Si l'agent B accepte, il cède de l'énergie à l'agent A. En revanche s'il refuse, il y a conflit, un « combat simulé » a lieu, chaque agent perd une quantité d'énergie proportionnelle au niveau d'énergie de l'agent opposé.

4.2.8 Mise à jour de l'« ardeur »

L'agent peut modifier son comportement lorsqu'il reçoit un ordre particulier, provenant de l'analyse ou de la morphologie. L'ardeur de l'agent est une variable qui est mise à jour par l'agent lorsqu'il reçoit une demande d'adaptation de son comportement. Si on demande à l'agent d'avoir plus d'énergie, il deviendra davantage ardent : la probabilité qu'il fasse une demande au cycle suivant augmente. En revanche, si on lui demande de disposer de moins d'énergie, il deviendra moins ardent : la probabilité qu'il fasse une demande d'énergie au cycle suivant diminue.

4.2.9 Boucle de simulation

Dans la boucle de simulation, tous les agents sont synchrones. À chaque pas de temps, chaque agent est exécuté et à la fin du cycle, l'envoi des messages et la mise à jour des états des agents (et éventuellement leur retrait pour cause de mort, ou leur ajout par clonage) sont effectués. À chaque cycle, tous les agents ont la même représentation de l'environnement.

4.2.10 But : convergence, maintien du niveau de la population

Le but du système est de maintenir le niveau de la population d'agents en la faisant converger vers une certaine valeur, et ceci en adaptant le comportement des agents grâce à la morphologie. Plus précisément, la convergence aura lieu sur un intervalle, valeur visée avec une marge.

4.2.11 Organigramme du comportement de l'agent

La figure 4.2.3 résume le comportement de l'agent aspectuel sous forme d'organigramme.

Le cycle de simulation pour un agent commence, sur l'organigramme, à « début ». Si l'agent a suffisamment d'énergie (niveau d'énergie supérieur à un seuil donné), celui-ci a la possibilité de se cloner. Le test du clonage consiste en une comparaison d'une variable aléatoire à une seuil. Si la valeur est supérieure, l'agent se clone.

L'agent a ensuite la possibilité d'initier une demande d'énergie à un autre agent. Ce choix est basé aussi sur une variable aléatoire comparée à un seuil fonction de son ardeur. Plus un agent est ardent, plus il a de chance à initier une demande d'énergie. Le choix de l'agent à qui est adressé la demande se fait de façon aléatoire.

Dans le cas où l'agent avait fait une demande d'énergie au cycle précédant, celui-ci est en attente de réponse. La réponse est soit positive, soit négative. Dans

le cas d'une réponde positive l'agent initiateur de la demande gagne de l'énergie, et l'autre agent en perd. Dans le cas d'une réponse négative, les deux agents perdent de l'énergie.

L'agent peut initier à son tour une demande d'énergie. Cela se fait par une comparaison d'une valeur aléatoire par rapport à un seuil fonction de son ardeur.

Et finalement, l'agent teste son niveau d'énergie. Si le niveau est positif, l'agent est maintenu dans l'organisation et il pourra effectuer un autre cycle de simulation. En revanche, si le niveau est nul ou négatif, l'agent est mort et il est retiré de l'organisation.

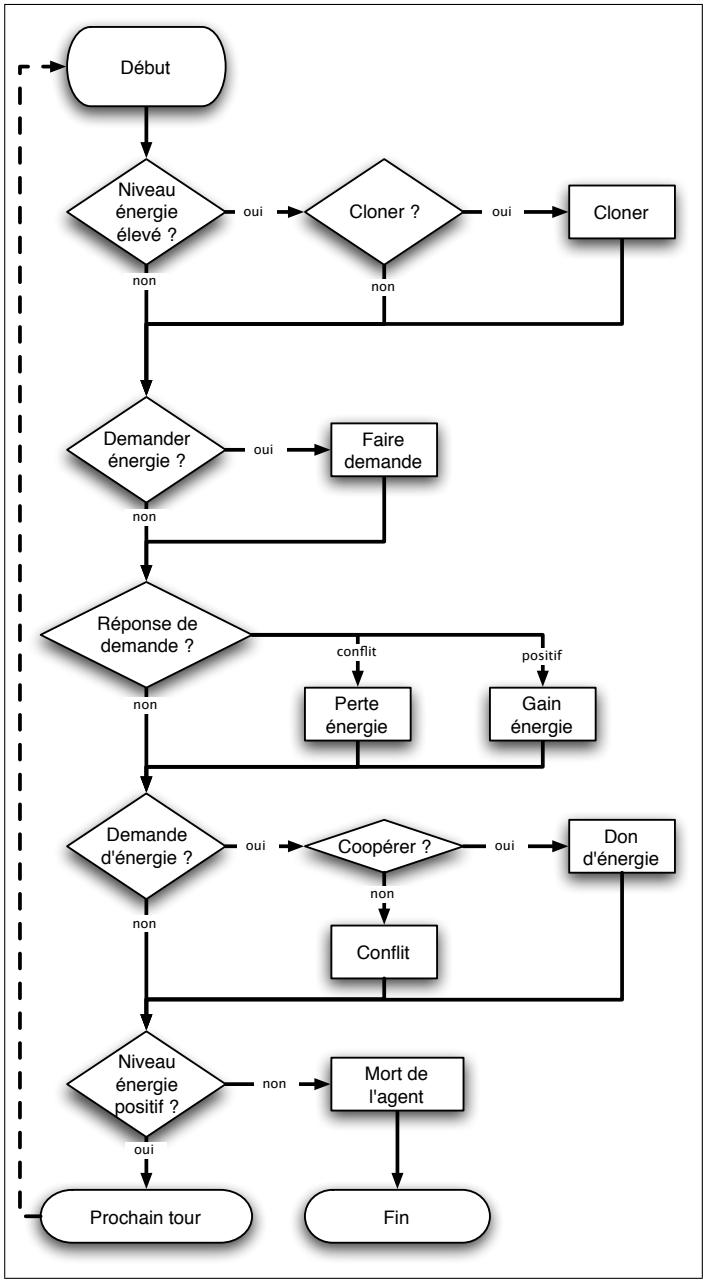


Figure 4.2.3 – *Organigramme du comportement de l'agent.*

4.3 Description de la méthode d'analyse et de contrôle

Dans cette section nous allons voir de quelle façon le système procède à la construction de la représentation de son état et de la mise en correspondance de l'état micro (l'état local des agents) avec l'état macro (l'état global de l'organisation) et, enfin, comment le pilotage s'effectue.

4.3.1 Choix de la mesure sur les agents

Dans cet exemple, pour rendre compte de l'état du système, une seule mesure locale est faite au niveau de l'agent. Cette mesure est tout simplement le niveau d'énergie de l'agent. Celle-ci est assez représentative de l'état organisationnel de l'agent : plus un agent a d'énergie, plus il a de chance de survivre et également plus il a de chance d'influencer l'organisation.

4.3.2 Choix de la forme

Il est assez naturel de rendre compte de l'état général de l'organisation par la distribution de cette mesure de l'organisation à l'aide d'histogrammes. À la fin de chaque cycle de simulation de l'organisation aspectuelle, la mesure du niveau d'énergie des agents est effectuée. Le module de morphologie recueille cette information et calcule la distribution de l'énergie. L'histogramme ainsi obtenu représente de manière « absolue » la distribution de l'énergie au sein de l'organisation. Cependant, nous estimions que cette distribution absolue n'était pas bien représentative de l'état de l'organisation et qu'une interprétation relative de cette distribution serait plus pertinente, et ce d'autant plus que le nombre d'agents peut varier, ainsi que l'énergie totale contenue dans l'organisation. Les histogrammes calculés sont ainsi centrés sur la valeur moyenne et normalisés : la surface est donc égale à l'unité (voir figure 4.3.1). De tels histogrammes permettent de rendre compte des disparités au sein de l'organisation de façon relative.

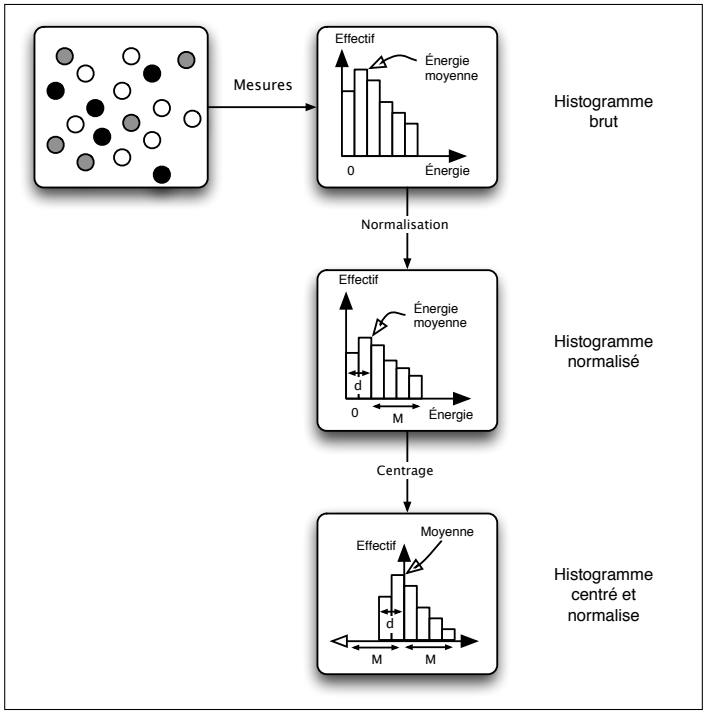


Figure 4.3.1 – *Centrage et normalisation de la forme (histogramme). La distribution est normalisée (aire égale à l'unité) et ensuite centrée autour de la moyenne. Le module d'analyse travaille sur ces formes normalisées et centrées.*

Le nombre d'intervalles (nombre de classes de l'histogramme), relativement petit, est fixe au cours d'une simulation. Il ne faut pas que le nombre d'intervalles soit trop important au risque de n'avoir que très peu d'agents par classe, ou trop faible au risque de ne pas bien distinguer les agents entre eux.

Cependant, comme la population varie en taille, il arrive que des histogrammes soient calculés à partir d'un petit effectif d'agents, ce qui produit des histogrammes non représentatifs de l'organisation. Ce type d'histogrammes génère du bruit dans l'ensemble d'apprentissage. Pour y remédier, les histogrammes dont l'effectif est inférieur à un certain nombre sont filtrés, ils sont ignorés par l'analyse.

4.3.3 Comparaison de formes

Les histogrammes ont l'avantage de pouvoir être aisément comparés entre eux et il est facile de proposer un ensemble de manipulations pour opérer sur les agents afin de conformer l'organisation. Cette manipulation est basée sur une permutation des éléments de l'histogramme. La méthode est décrite ci-dessous (section 4.3.4).

Distance

Pour comparer des histogrammes entre eux, il suffit de les considérer comme des vecteurs, d'utiliser une norme classique et de comparer la distance à une valeur seuil. Avec un peu de formalisme mathématique, on peut écrire : soit deux histogrammes $\Phi_1 = (\phi_1^1, \phi_1^2 \ldots \phi_1^n)$ et $\Phi_2 = (\phi_2^1, \phi_2^2 \ldots \phi_2^n)$. On définit la distance $d(\Phi_1, \Phi_2)$ par : $\sqrt{\sum_{i=1..n} (\phi_1^i - \phi_2^i)^2}$. Cette métrique, arbitraire mais usuelle , a le mérite d'être simple et rapide à calculer. Une méthode plus sophistiquée pourrait s'inspirer de la méthode du χ^2 par exemple.

Formes équivalentes

On définit une relation d'équivalence entre les histogrammes de façon classique, en comparant la distance entre deux histogrammes à une valeur seuil. On dira que deux formes (ici, des histogrammes) Φ_1 et Φ_2 sont ε-équivalentes si $d(\Phi_1, \Phi_2) < \varepsilon$ (avec ε positif).

4.3.4 « Réciproque »

Afin de guider le système, il faut modeler, contraindre, sa structure. Pour cela, il faut pouvoir donner les ordres adéquats aux agents pour qu'ils modifient leurs comportements de sorte que la structure générale du système tende vers la forme recherchée. Dans le cas présent, il s'agit de comparer deux distributions, l'histogramme courant de l'organisation et l'histogramme souhaité, et de calculer les

modifications à apporter à la forme courante pour faire tendre la forme de l'organisation vers celle qui est recherchée. Ceci est illustré sur la figure 4.3.2. Une approximation peut être donnée en se basant sur la permutation des éléments dans les classes de l'histogramme.

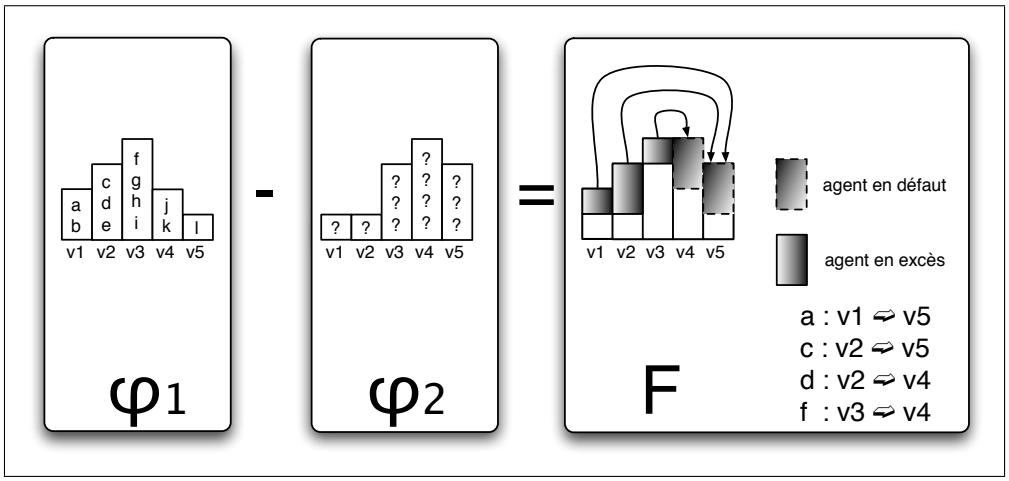


Figure 4.3.2 – *Transformation d'un histogramme : la forme Φ_1 est la forme courante, Φ_2 la forme recherchée. Les agents a, c, d et f sont en excès dans les classes v_1, v_2 et v_3. Des ordres leur sont donnés pour qu'ils essaient de rejoindre les classes v_5 et v_4 respectivement.*

Soit Φ_1 la forme courante de l'organisation et Φ_2 la forme recherchée. On effectue la différence classe par classe de Φ_1 et Φ_2. On détermine quels sont les agents qui sont en excès dans Φ_1 par rapport à Φ_2, et quelles sont les classes auxquelles il manque des agents. Des ordres leur sont donnés pour qu'ils tentent de rejoindre les classes adéquates afin changer la forme de l'organisation. Les ordres envoyés aux agents sont en fait des injonctions qui jouent sur leur paramètre d'ardeur. Le mécanisme d'adaptation du comportement de l'agent est assez simple, et il est fondé sur l'hypothèse suivante : il y a une corrélation positive entre l'ardeur et l'énergie de l'agent. Ainsi, si l'on envoie un ordre à l'agent pour qu'il soit dans une classe supérieure (en terme d'énergie), l'ardeur de l'agent sera augmentée (la probabilité pour qu'il demande de l'énergie à d'autres agents augmente), et inversement pour les classes inférieures.

4.3.5 Corrélation de formes et de comportement du système

Classification élémentaire

Le rôle du module d'analyse est d'apprendre et de classer les formes calculées par la morphologie. Ce module connaît également la taille de la population. Cette donnée lui est transmise à chaque cycle. De cette façon, le module d'analyse peut facilement déterminer si le système agit « bien » ou « mal ». Le but, rappelons le, est que la taille de la population converge autour d'une valeur donnée (voir figure 4.3.3) :

- le système se comporte de façon adéquate si la population converge vers cette valeur (plus exactement se maintienne dans un intervalle) ;
- sinon, en cas de divergence, on dira que le système se comporte mal.

Nous appellerons ces règles de classification des règles binaires.

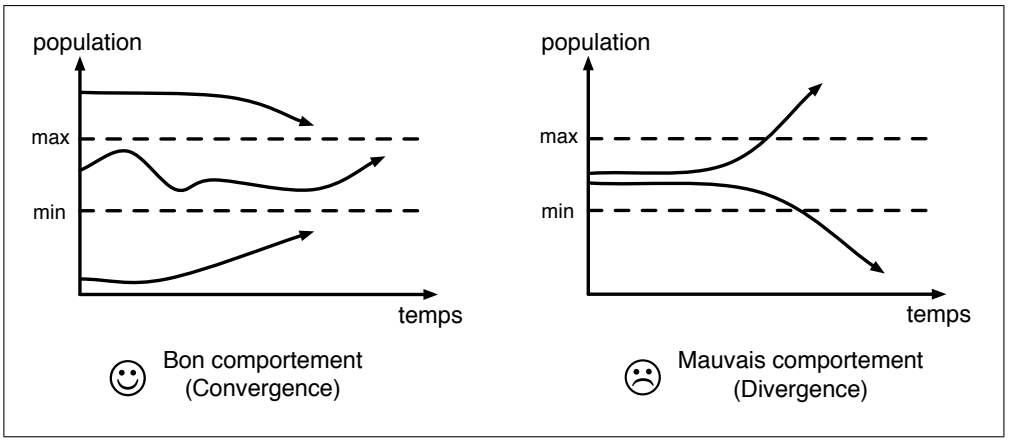


Figure 4.3.3 – *Règles élémentaires utilisées pour classer le comportement du système et les formes.*

Classification plus fine

Une discrimination plus fine par rapport aux règles élémentaires peut également être considérée (voir la figure 4.3.4) :

- en tenant compte de la pente moyenne de la courbe d'évolution de la population (pente quasi nulle, pente positive, pente négative)
- et regardant si le niveau de la population se trouve dans l'intervalle, ou bien au-dessus ou en dessous.

Nous appellerons ces règles de classification les règles ternaires.

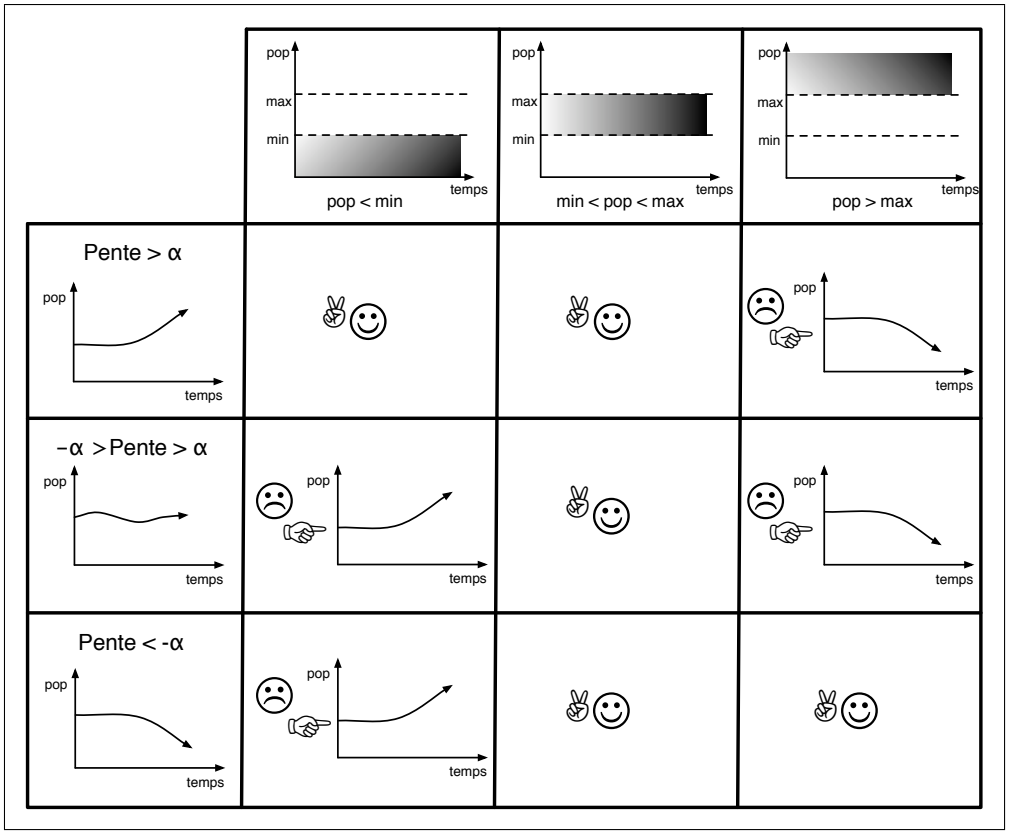


Figure 4.3.4 – *Règles, classifications plus fines et corrections à apporter selon le cas de divergence.*

Le comportement de l'organisation sera considéré comme adéquat dans les cas suivants :

- le niveau de population est compris dans l'intervalle, quel que soit la pente ;
- le niveau de population est trop faible, mais la pente est positive ;
- le niveau de la population est trop élevé, mais la pente est négative.

Dans tous les autres cas, le comportement sera considéré comme inadéquat.

4.3.6 Apprentissage et retour sur le système

Dans cet exemple, lorsque le système se comporte convenablement, lorsqu'il y a convergence de la population, (voir figure 4.3.5), il n'y pas de retour de la part de l'analyse (et donc, non plus de la morphologie). L'organisation aspectuelle évolue librement, le module de morphologie construit la représentation organisationnelle de l'organisation aspectuelle, cette représentation est donnée au module d'analyse qui, tout simplement, classe la forme produite comme étant bonne.

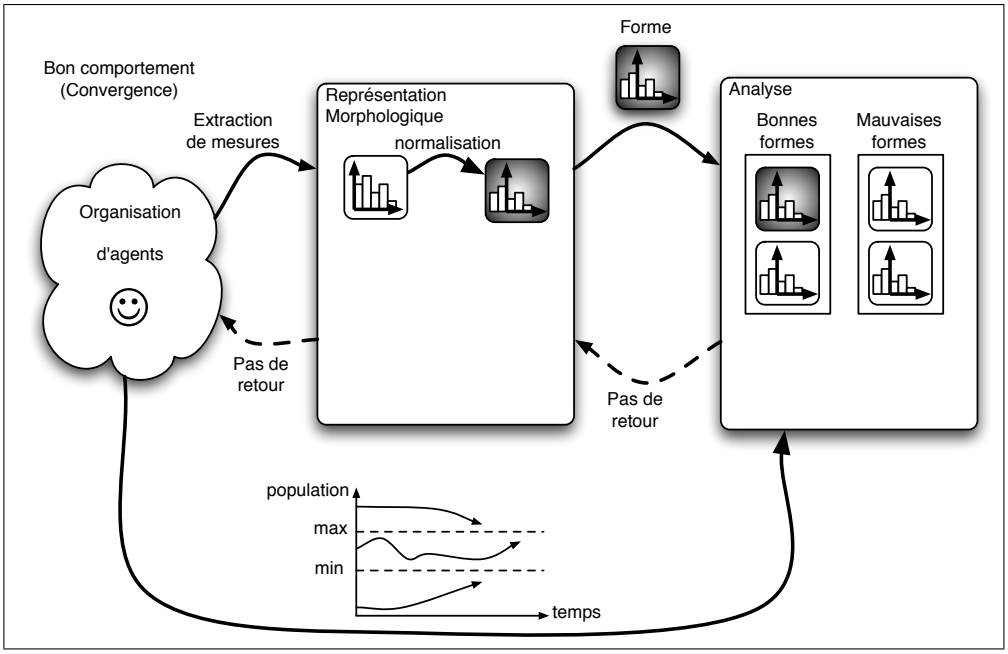


Figure 4.3.5 – *Lorsque le système se comporte bien, il n'y a pas de retour.*

En revanche, lorsque la population diverge (figure 4.3.6), l'analyse détecte alors l'anomalie (grâce aux règles fournies par le concepteur) et entre en jeu : elle passe par la morphologie pour corriger le comportement du système : le module d'analyse choisit une « bonne » forme parmi celles qu'il a apprise, si possible proche de celle reflétant l'état actuel de l'organisation (en utilisant la distance entre les formes). Dans cet exemple, l'analyse n'opère pas directement sur l'organisation aspectuelle, elle passe par la morphologie. Lorsque la morphologie reçoit un ordre du module d'analyse, le module de morphologie calcule les ordres à donner aux agents aspectuels pour modifier la structure de l'organisation (comme indiqué à la section 4.3.4).

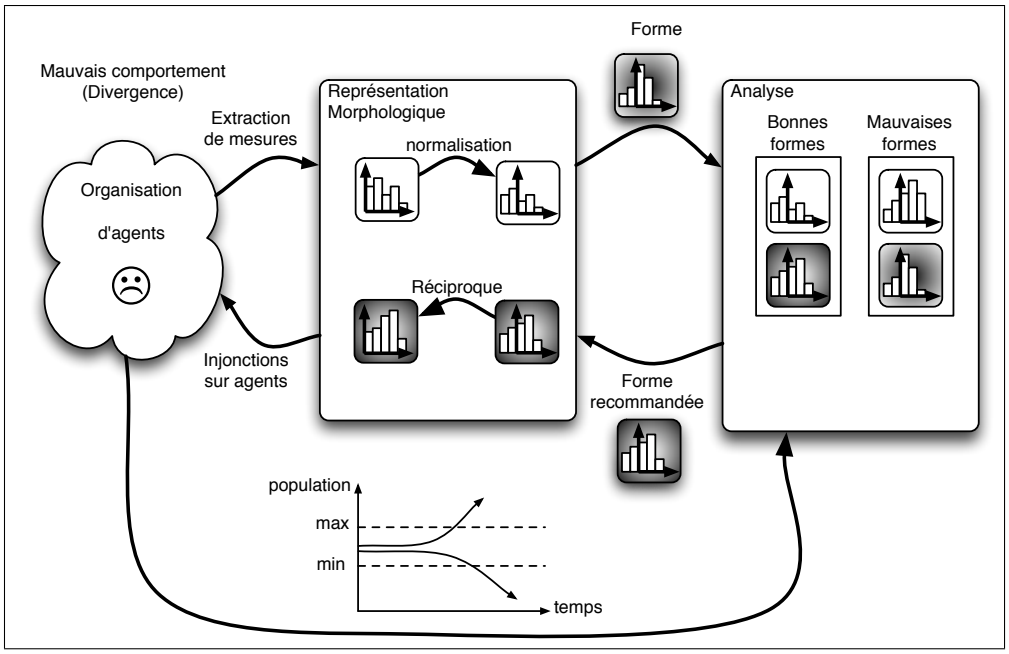


Figure 4.3.6 – *Lorsque la population diverge, le système opère un contrôle par la morphologie.*

4.4 Expériences et résultats

Le système a été implanté sur machine en O'Caml [oca] (sur une machine de bureau, et également sur portable). Les expériences ont consisté à lancer l'exécutable avec différents paramètres un nombre représentatifs de fois afin d'évaluer le comportement moyen du système.

4.4.1 Test de l'hypothèse

Nous avons d'abord vérifié que des formes spécifiques choisies comme attracteur jouaient effectivement leur rôle d'attracteur, et que la convergence du système n'était pas due à « l'excitation » des agents, ou à la perturbation de l'organisation, en leur envoyant des messages. Nous avons modifié le système comme suit : au lieu que le module d'analyse choisisse une bonne forme pour corriger le système lorsque son comportement fait défaut, la forme envoyée à la morphologie comme attracteur est générée au hasard (un histogramme normé mais généré aléatoirement). Nous avons effectué des tests avec ce procédé. Nous avons observé que le contrôle était inexistant. Le système se comportait de façon pire que sans mécanisme de contrôle, la population divergeait plus souvent. Nous pouvons conclure que le bruit ne peut permettre de faire converger le système.

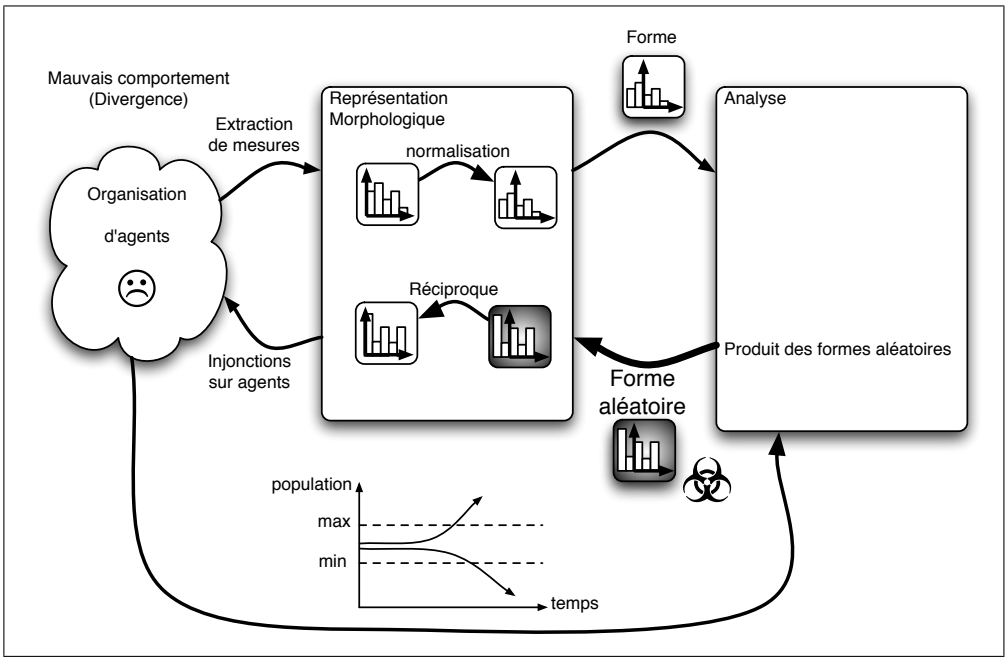


Figure 4.4.1 – *Test de l'hypothèse. Dans ce cas, les formes envoyées à la morphologie par le module d'analyse en cas de divergence sont produits aléatoirement. Avec ce type de formes, le contrôle est inexistant.*

4.4.2 Quelques résultats

Afin d'évaluer le comportement moyen du système, nous avons effectué une série de tests. La figure 4.4.2 illustre quelques comportements typiques du système, sans et avec contrôle, pour une population cible de 50 agents, avec un intervalle de 40 à 60. « Sans contrôle » signifie que l'organisation aspectuelle était livrée à elle-même.

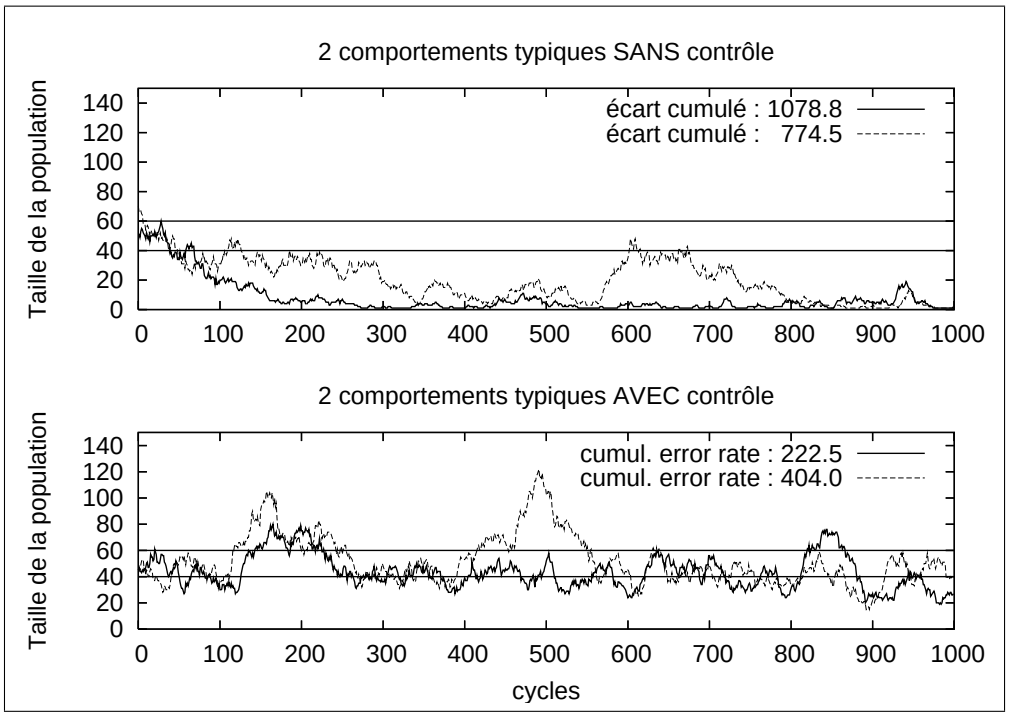


Figure 4.4.2 – *Exemple de courbes de population représentant le comportement typique du système avec et sans contrôle.*

Une grande série de tests a montré que cette organisation tendait à diverger dans la très grande majorité des cas. En revanche, avec le contrôle, la population d'agents était maintenue autour de cet intervalle. Comme le révèle le graphe du bas de la figure 4.4.2, le niveau de la population n'est pas exactement maintenu dans l'intervalle, cependant le niveau oscille et converge autour de cet intervalle. Ceci peut être expliqué par le fait que le mécanisme de contrôle ne s'enclenche qu'une fois que le niveau de population sort de l'intervalle. D'autre part, la convergence relative peut être expliquée par la volatilité d'un tel système : il se peut que

beaucoup d'agents soient clonés en même temps lors d'un même cycle (la population peut théoriquement doubler d'un cycle à l'autre). Sur la figure 4.4.2, on peut y lire l'écart cumulé de chaque courbe par rapport à l'intervalle cible (intégrale du carré de la distance de l'intervalle) : respectivement $1078,8$ et $774,5$ sans contrôle ; et 222.5 et 404.0 avec contrôle.

La figure 4.4.3 montre un comparatif entre le système avec et sans contrôle pour une population de taille ciblée à 50 agents (plus ou moins $0,2$). Ce comparatif indique la distribution des écarts (les valeurs d'écart des courbes comme celles indiquées sur la figue 4.4.2) pour un grand nombre de simulations. L'abscisse représente l'écart de la courbe du niveau de population. On peut dire que globalement, l'erreur avec le mécanisme de contrôle est diminuée, la courbe avec contrôle est centrée sur 200–300, alors que la courbe sans contrôle est centrée sur 700–800 (si on considère ces distributions comme suivant la loi normale). Cependant la courbe avec contrôle est davantage étalée. La courbe des cumuls permet de mettre en évidence que dans près de 90 % des cas le système de contrôle permet à la population de mieux converger. Dans 10 % des cas environ, le mécanisme de contrôle fait diverger le système davantage. Ceci peut être expliqué par le mécanisme d'apprentissage qui peut être mis en défaut au début de la simulation. Si le système se trouve dans une conformation divergente dès le début, les formes qu'il apprend sont représentatives d'un comportement de divergence. Il se peut que le système ne se présente pas dans un état de convergence au début. Dans ce cas, le module d'analyse ne pourra proposer de forme adéquate. C'est comme si le module d'analyse n'existait pas. Nous avons vu qu'en l'absence de module de d'analyse le comportement moyen de l'organisation aspectuelle est de diverger.

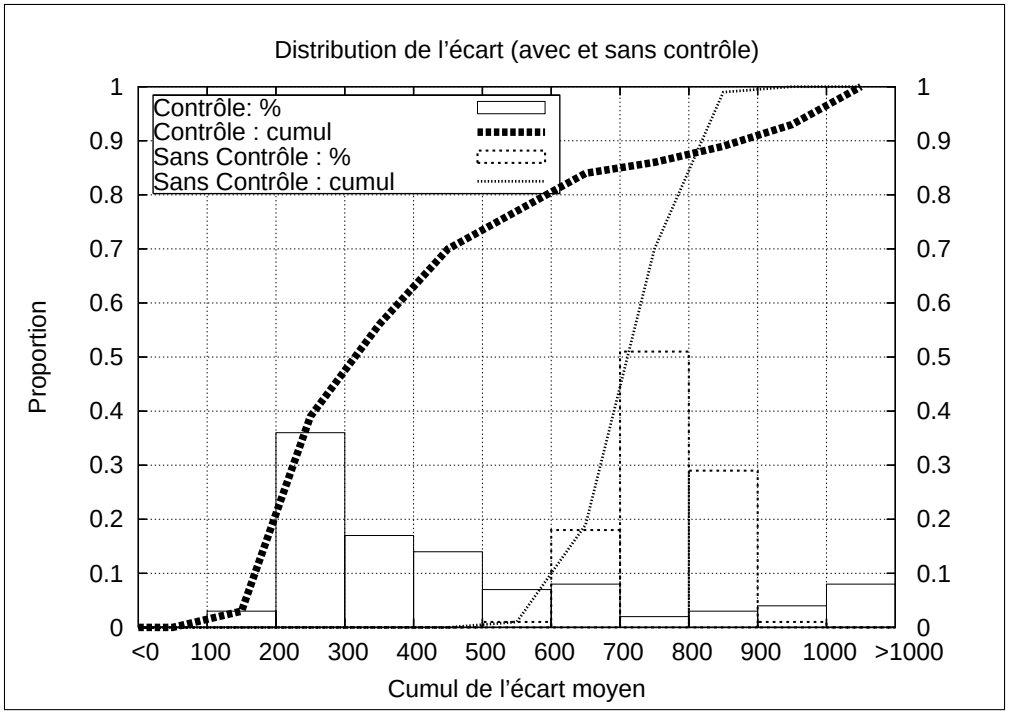


Figure 4.4.3 – *Distribution de l'écart de la population sur plusieurs expériences, avec et sans contrôle.*

4.4.3 Étude des paramètres

Nous avons effectué plusieurs tests pour voir l'effet de la variation des paramètres sur la qualité de contrôle. Les paramètres que nous avons étudiés sont les suivant :

- le nombre d'agents cible (50, 100, 200, 300, 400, 500 et 600) ;
- la marge d'erreur pour la convergence ($0,10$, $0,05$ et $0,01$) ;
- la densité des histogrammes (3, 4, 5, 7 et 10).

Pour chacune des combinaisons de paramètres, nous avons effectué 10 tests, chacun avec 10 organisations d'agents différentes. En d'autres termes, pour chaque combinaison de paramètres, 10 modules d'apprentissage différents ont été exposés à 10 organisations à la suite pendant 2000 cycles de simulations. L'ensemble d'apprentissage du module d'analyse est vierge lorsqu'il est exposé pour la pre-

mière des 10 organisations, mais lorsqu'il est exposé aux autres organisations, il retient les formes qu'il a obtenu précédemment.

Le résultat des expériences des différentes combinaisons de paramètres est résumée dans les figures numérotées de 4.4.4 à 4.4.9. Ces graphes se lisent par paires. Ces deux types de graphes, distribution et cumul, ont été séparés pour facilité la lecture des courbes. Ces courbes proposent de voir la répartition du rapport entre la valeur moyenne de la population au cours d'une simulation v_m et la valeur cible v. De façon sommaire, plus la distribution du rapport $\frac{v_m}{v}$ se rapproche et se centre autour de 1, plus le contrôle est bon. Ainsi les combinaisons de paramètres qui font tendre les courbes autour de l'abscisse 1 sont meilleurs.

Étude de la variation de la densité

Les courbes 4.4.4 et 4.4.5 révèlent que le choix de la forme pour représenter la structure de l'organisation joue un rôle important sur la performance du système. Plus exactement, dans ce cas, une résolution trop basse des histogrammes (c'est à dire le nombre de classes ou la densité des histogrammes est faible) influe très fortement sur le comportement du système. En effet, on observe que pour des histogrammes de 3 et 4 classes, le comportement moyen du système est relativement divergeant sur l'ensemble des autres combinaisons de paramètres. En revanche, pour des densités supérieures à 5 (c'est-à-dire de 5, 7 ou 10), le comportement moyen du système s'améliore nettement et les distributions du rapport $\frac{v_m}{v}$ sont centrées autour de 1 et adoptent à peu près la forme d'une distribution normale. Cette observation indique qu'une forme trop grossière (histogramme à peu de classes) ne permet pas au système de bien distinguer les différentes structures le conduisant à discriminer les bonnes et mauvaises formes autorisant un bon pilotage du système. On peut également observer que la classification fine (figure 4.4.5) améliore quelque peu la performance du contrôle.

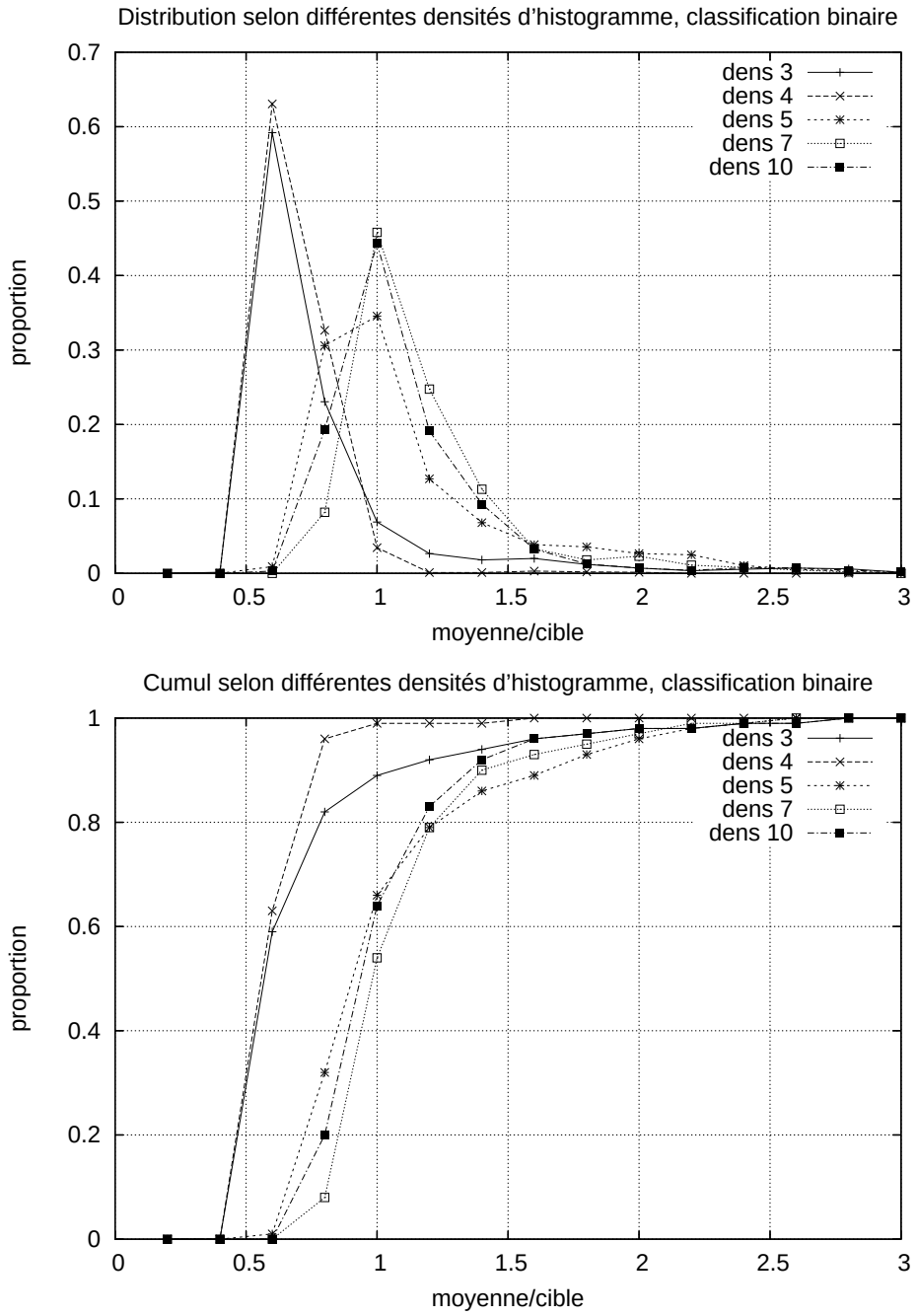


Figure 4.4.4 – Distribution de $\frac{v_m}{v}$, règles binaires, en fonction de la densité des histogrammes.

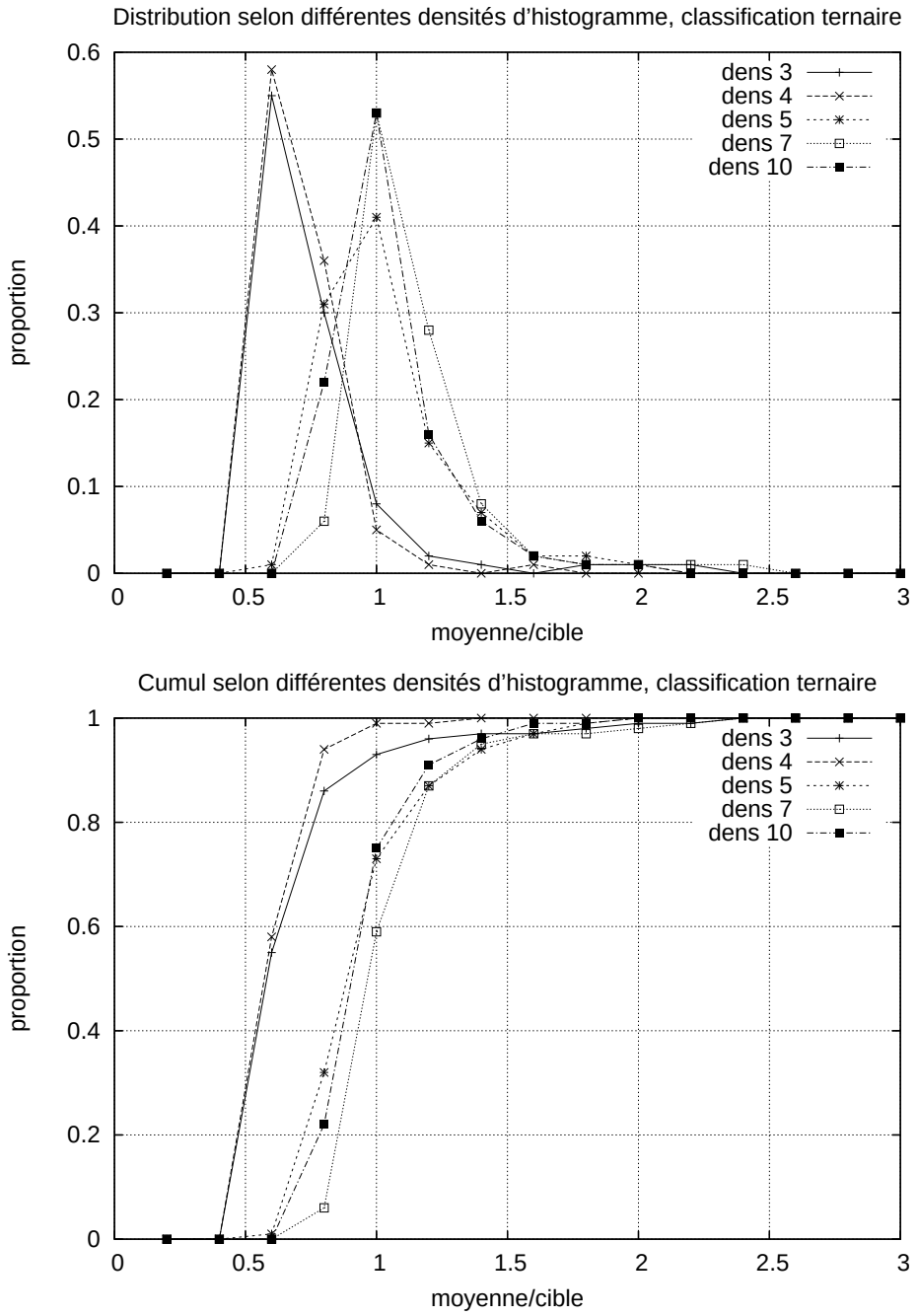


Figure 4.4.5 – Distribution de $\frac{v_m}{v}$, règles ternaires, en fonction de la densité des histogrammes.

Étude de la variation de la marge

L'étude du paramètre de marge (figures 4.4.6 et 4.4.7) révèle que globalement plus la marge est élevée, plus la performance est meilleure. Ceci peut être expliqué de deux façons qui sont liées. D'une part, d'une manière mécanique : plus la marge est grande, plus l'intervalle ciblé est grand, et moins l'écart possible est grand. D'autre part, plus la marge est grande, plus l'ensemble des bonnes formes est potentiellement grand. On peut dire que le problème du paramètre de la marge est davantage lié à un problème d'apprentissage et de découverte de bonnes formes. Un mécanisme d'apprentissage faisant intervenir une marge rétrécissante au cours du temps aurait sans doute eu de meilleures performances. On peut noter deux populations sur ces courbes (deux distributions qui se traduisent par deux pics) qui correspondent aux deux populations observées dans les graphes précédents analysant l'effet de la précisions des histogrammes (Courbes 4.4.4 et 4.4.5).

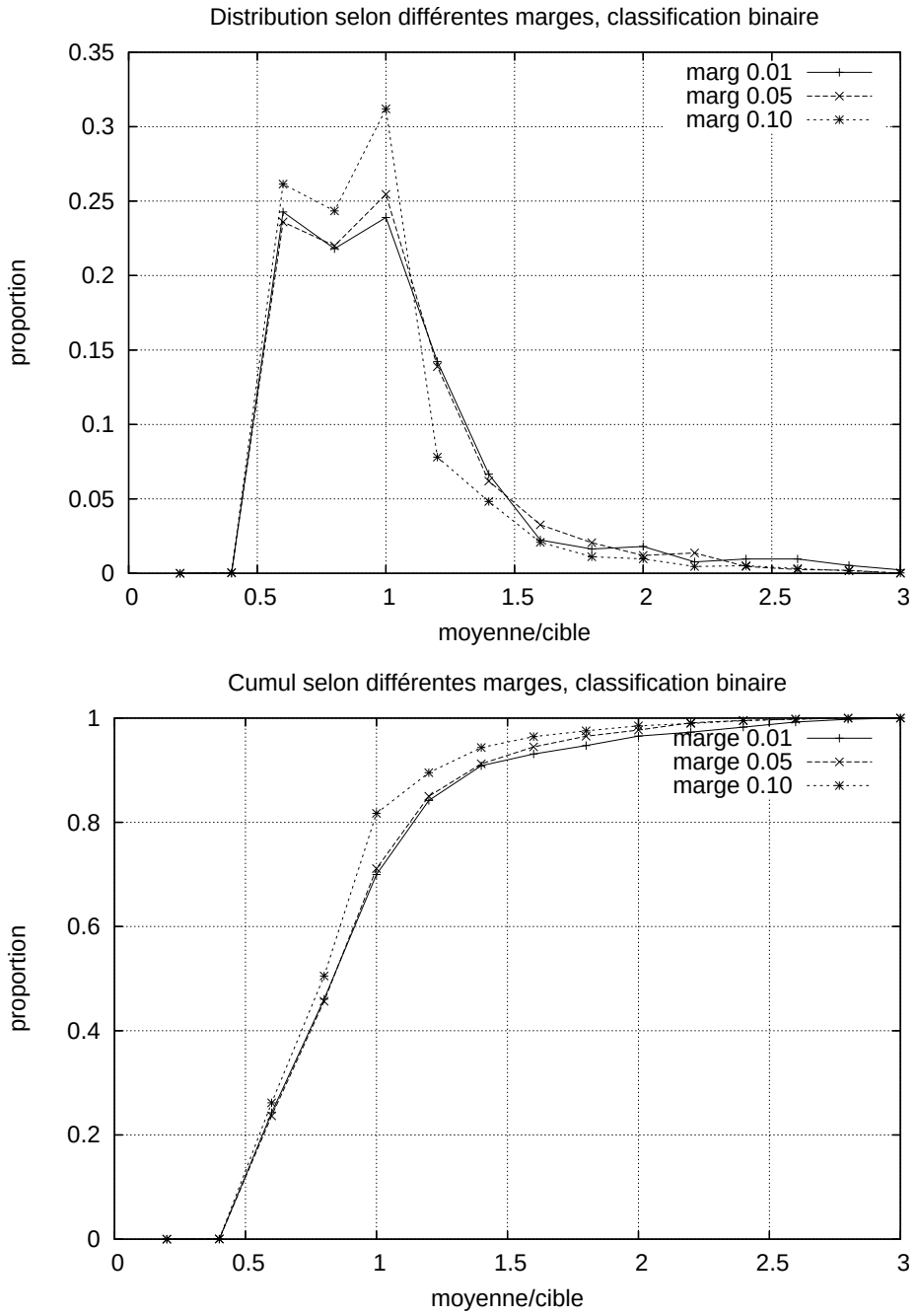


Figure 4.4.6 – Distribution de $\frac{v_m}{v}$, règles binaires, en fonction de marges différentes.

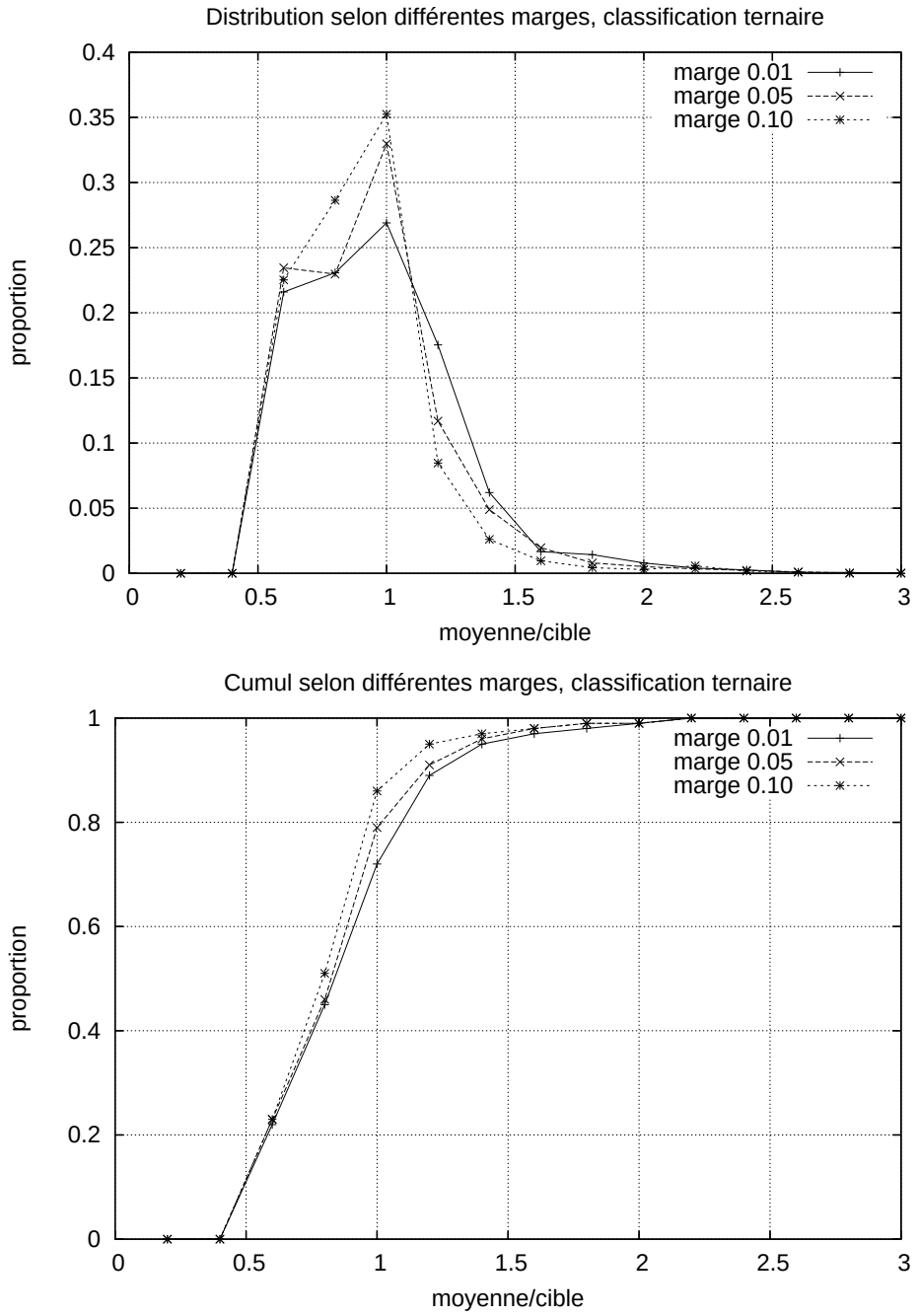


Figure 4.4.7 – Distribution de $\frac{v_m}{v}$, règles ternaires, en fonction de marges différentes.

Étude de la variation de la taille

L'étude de la variation de la taille de la population montre, sur les figures 4.4.8, 4.4.9, que le contrôle est meilleur, et d'autant plus que la taille visée de la population est grande. Cependant il faut bien tenir compte du phénomène de lissage qu'un grand nombre d'agents implique. En effet, un même écart de taille sera d'autant moins perceptible que la taille de la population visée est plus grande. Néanmoins pour des tailles de populations d'agents du même ordre de grandeur (200 à 600), on note une certaine homogénéité dans la forme des distributions. On perçoit également deux populations distinctes sur ces courbes (deux pics différents), qui correspondent aux deux populations différentes de mauvais et bon contrôle, dû aux différentes densités des histogrammes. On peut également noter que la classification fine améliore un peu la performance du contrôle.

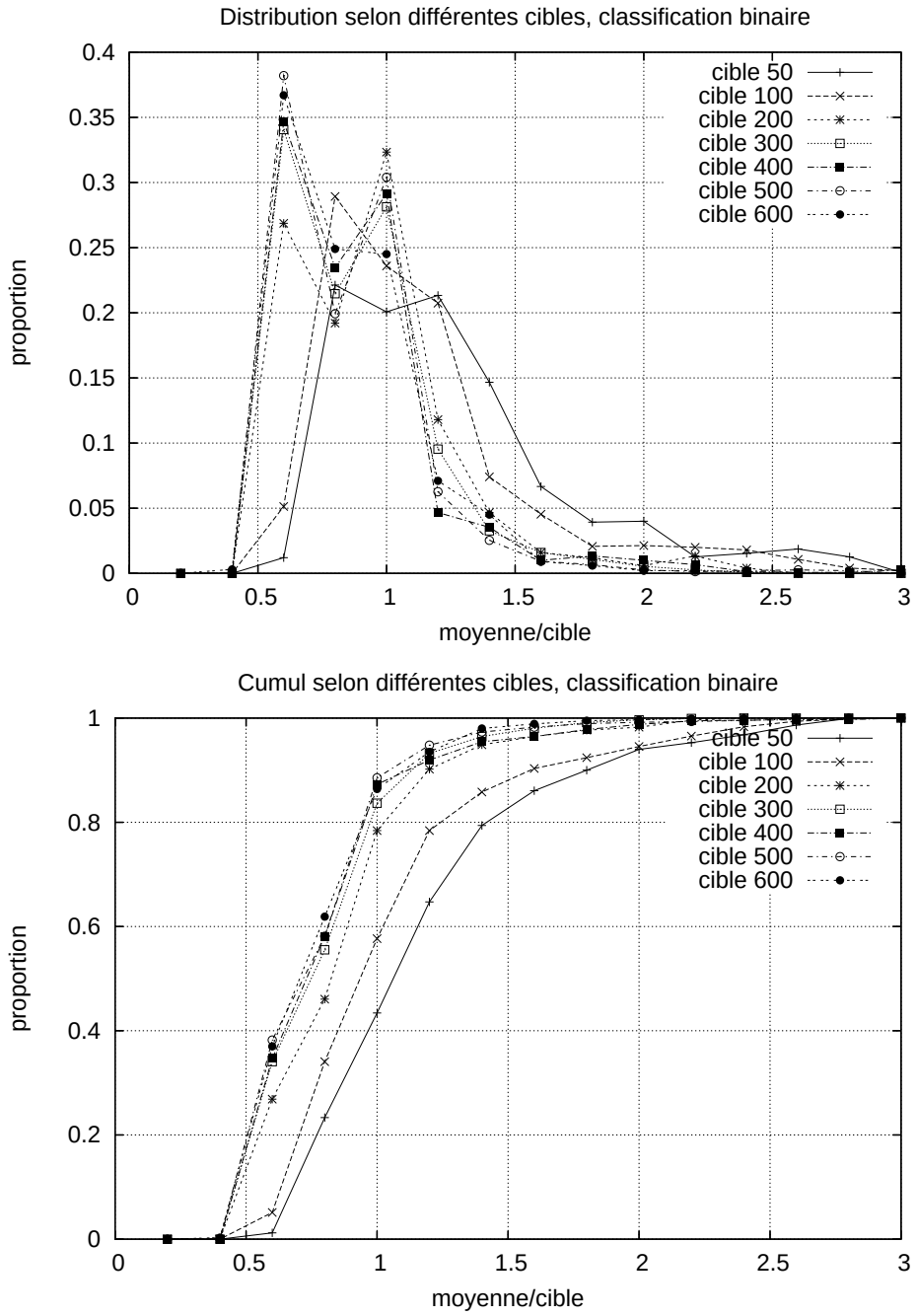


Figure 4.4.8 – Distribution de $\frac{v_m}{v}$, règles ternaires, en fonction de cibles différentes.

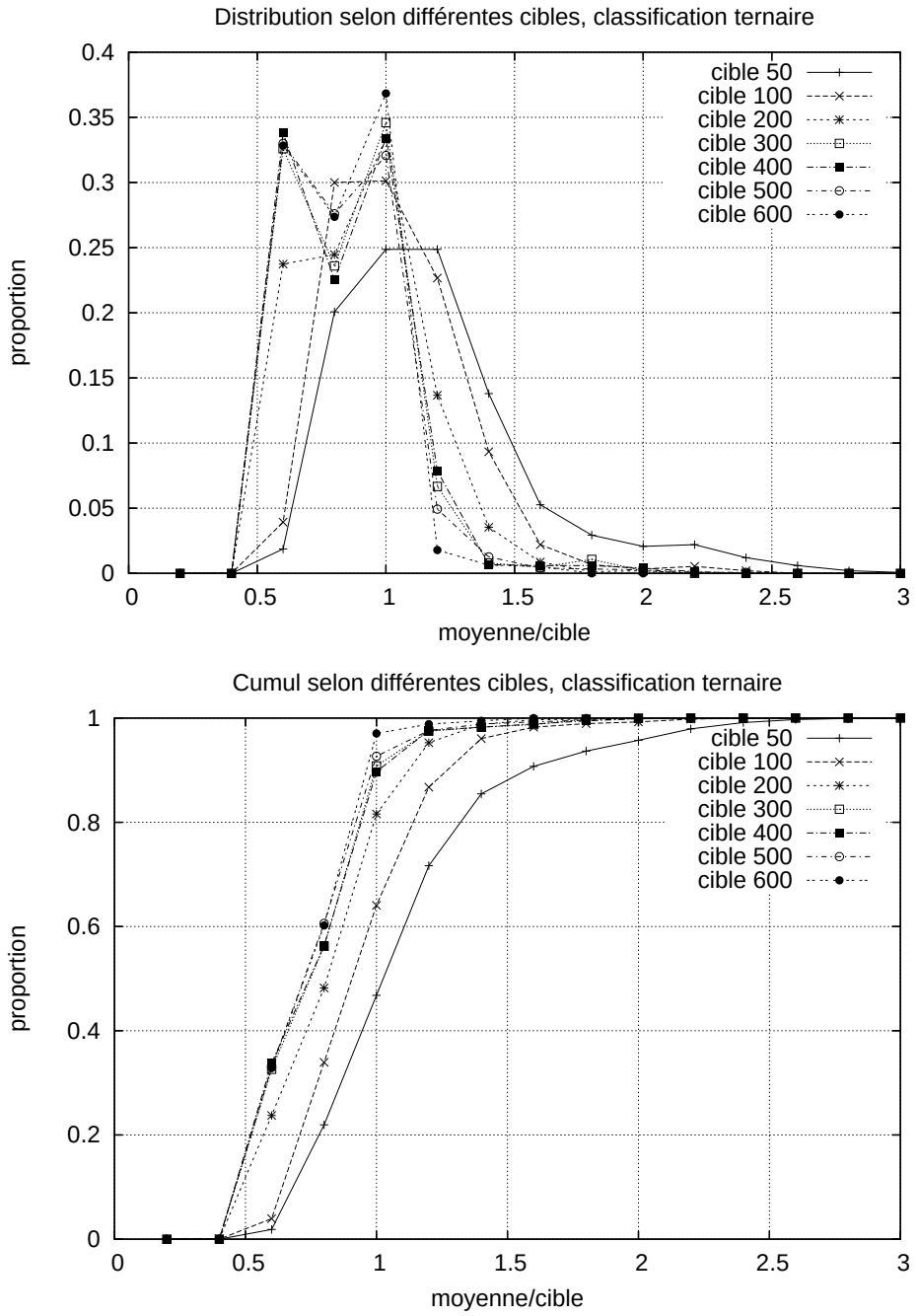


Figure 4.4.9 – Distribution de $\frac{v_m}{v}$, règles ternaires, en fonction de cibles différentes.

Remarques

De façon synthétique, on peut observer après une première analyse que lorsqu'il n'y a pas beaucoup d'agents et que les histogrammes sont trop grossiers, le système ne converge pas très bien. Ceci est peut-être dû à une faible signification des histogrammes pour des populations trop faibles. Malgré la dispersion des courbes, on peut noter une certaine homogénéité des courbes : les courbes de distributions et de cumul ont sensiblement la même forme selon les différentes valeurs des paramètres, à l'exclusion peut-être des courbes pour les valeurs faibles de densité des histogrammes, qui ne permettent pas de discrimination suffisante pour bien piloter le système.

4.4.4 Formes apprises

La figure 4.4.10 regroupe l'essentiel de l'ensemble d'apprentissage pour des modules d'analyse différents pour un cas de paramètres (300 agents, marge $0,1$ et densité de 10). En première approximation, on peut constater qu'il y a globalement deux formes principales similaires : forme A et forme B (les autres formes ont été négligées). Le taux de réussite de contrôle de chacun de ces modules est très fortement corrélé à la fréquence d'apparition élevée d'une des formes (la forme A) dans l'ensemble d'apprentissage (voir figure 4.4.11). Ces ensembles sont classés du meilleurs au pire.

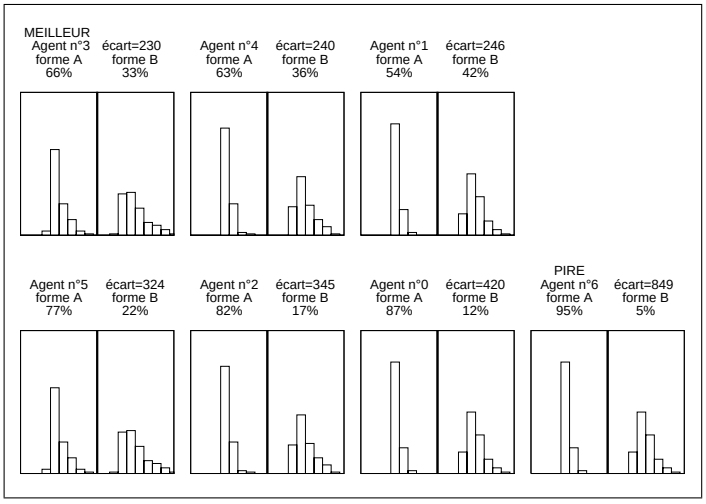


Figure 4.4.10 – *Formes principales des ensembles d'apprentissage dans différents cas. Deux formes sont retenues en général. Les paires de formes sont relativement similaires. Elles sont classées du meilleur cas au pire des cas.*

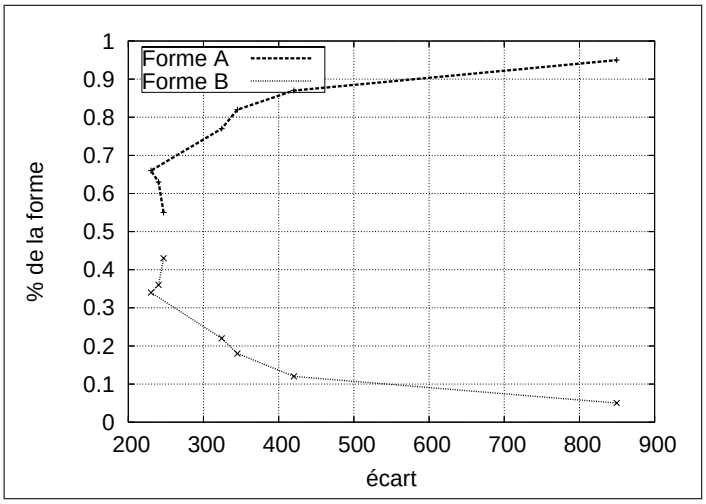


Figure 4.4.11 – *Graphe représentant la proportion de la forme apprise en fonction de l'écart moyen.*

4.5 Conclusion

Dans cette partie nous avons décrit un exemple d'application du modèle de contrôle de population d'agents simples et plutôt réactifs. Nous avons décrit le fonctionnement des agents ainsi que le mécanisme de contrôle appliqué à cet exemple. Nous avons décrit quelques résultats illustrant l'influence des paramètres sur la qualité du contrôle. Bien que les paramètres influent sur le contrôle, nous notons une certaine homogénéité des courbes décrivant la qualité du contrôle (mis à part le cas extrême des faibles densités des histogrammes).

La principale limite de cet exemple est sa simplicité. Mais ne perdons pas de vue que cet exemple avait été développé pour valider certains aspects de notre modèle. La simplicité se trouve dans le comportement des agents. Ces agents sont réactifs. Leur comportement possède un seul degré de liberté (leur ardeur). Une seule mesure était effectuée sur les agents (leur niveau d'énergie). La simplicité de l'organisation étudiée ne permettait pas l'étude de structure plus compliquée que la distribution d'une valeur numérique au sein de l'organisation. Néanmoins cet exemple simple nous a permis de mettre au point certains aspects de notre approche et de proposer un exposé facile à appréhender en faisant le tour de la boucle de contrôle sans perdre de vue le mécanisme global du contrôle.

Chapitre 5

Agentification aspectuelle et prototype

Dans ce chapitre, nous présentons une méthode d'agentification, dite « agentification aspectuelle » par le biais d'un prototype de plate-forme exploitant cette méthode. Il s'agit d'une extension de travaux initiés par Alain Cardon [Car99]. Nous présentons cette méthode de façon générale (sections 5.1). Ensuite nous décrivons le prototype du système avec davantage de précision (section 5.2), notamment au sujet des agents aspectuels (section 5.3). Nous verrons qu'un tel système complexe devra être piloté (section 5.5). Ce type de système se prête bien à l'analyse morphologique que nous présenterons finalement (section 5.6). Dans le chapitre suivant (chapitre 6) nous présentons un cas d'application.

5.1 Introduction

L'agentification aspectuelle, initialement proposée par Alain Cardon [Car99] est une méthode d'agentification radicale. Elle repose sur la décomposition du problème que l'on souhaite étudier en aspects (ou concepts), qui peuvent être eux-mêmes décomposés en d'autres aspects, et qui sont réifiés toujours dans des agents. Cette méthode d'agentification se prête bien à la construction de système de contrôle de robot par exemple, car elle fut développée avec ce but, mais elle est applicable à d'autres types de problèmes.

Cette méthode d'agentification est dite « massive ». Elle est massive en terme de nombre d'agents et en terme de niveaux conceptuels. Pour un concept donné, il n'y aura pas un unique agent pour le représenter mais plusieurs, et ces agents n'auront pas tous exactement le même rôle, mais des rôles semblables, proches voire opposés. Par ailleurs, les agents n'exprimeront pas tous des rôles de même niveau conceptuel, certains auront des rôles de grain fin, ils s'intéresseront aux petits détails du problème, et d'autres auront des rôles plus globaux, comme la supervision du comportement général de parties dynamiques du système.

Nous allons donner un exemple d'école pour préciser ces notions, et qui sera repris ultérieurement pour illustrer les différentes autres notions. Considérons le problème simplifié de la reconnaissance de formes, comme par exemple celui d'un sous-module de vison d'un robot. Le robot pourra être amené à reconnaître différents types d'objets dans l'espace qui l'entoure. Certains agents pourront reconnaître directement des objets, d'autres ne s'intéresseront uniquement qu'à leur forme, d'autres à leur couleur. Par ailleurs nous supposerons que ces différents agents peuvent se tromper, involontairement ou volontairement, dans la reconnaissance. Ceci par exemple si les conditions d'éclairage ne permettent pas aux agents traitant la couleur de bien l'identifier, ou si le système est biaisé en donnant à celui-ci une légère sensibilité à une certaine gamme de couleurs ou d'objets. Chacun des agents interprétera un peu à sa manière l'environnement auquel il est exposé. La représentation de la scène extérieure est ainsi construite par l'organisation des agents aspectuels qui est en mouvement. Par ailleurs, les agents ont un comportement social, ils interagissent ensemble, ils peuvent former des groupes, des coalitions pour renforcer leur position et le trait de la scène qu'ils expriment, ou bien réduire d'autres agents qui réifient des aspects différents ou opposés, ou même similaires... Un trait de l'environnement sera alors représenté non pas de manière statique mais par l'activité et les mouvements engendrés par les agents. Par exemple, si la scène contient un arbre à feuilles vertes, certains agents s'activeront parce qu'ils auront reconnus la couleur verte, mais différents tons de vert pourront être également perçus. Selon le paramétrage des agents, tous les agents sensibles au vert pourront s'activer et coopérer et éventuellement se regrouper pour manifester cet aspect de la scène. Dans un autre cas de paramétrage, si la décomposition du rôle de reconnaissance du vert est fine, différentes gammes de vert pourront être perçues par différents agents qui pourront alors entrer en conflit. Par exemple, certains diront qu'il s'agit d'un vert clair, d'autre d'un vert foncé. D'autres part, pour donner un exemple d'agentification à différents niveaux conceptuels, l'organisation pourra avoir des agents qui s'intéresseront à des feuilles individuelles et d'autres agents s'intéresseront uniquement à des régions grossières de taches vertes correspondant au feuillage.

Cette introduction a présenté l'idée générale de notre approche. Dans la suite nous entrons dans les détails d'une réalisation d'un tel système. Il est clair qu'un tel système a peu de chances d'opérer avec son seul niveau aspectuel de façon satisfaisante. C'est pour cela qu'il sera proposé ultérieurement de piloter le système multi-agent par la morphologie. Cette méthode d'agentification a un double rôle : celui de proposer un système de contrôle d'un robot par exemple, et de servir de substrat d'étude à notre approche de systèmes multi-agents par la morphologie.

5.2 Description générale du prototype

Afin de pouvoir traiter différents problèmes, nous avons réalisé, sous forme de prototype, un système générique. Le système peut être considéré comme un moteur générique, dont l'application passe par l'établissement d'une ontologie du problème que l'on veut traiter. L'établissement d'une ontologie consiste essentiellement à déterminer les objets que les agents manipulent, les rôles des agents et de quelle manières ces rôles interagissent.

Nous commencerons par présenter les différents éléments du prototype (section 5.2.1), le système de communication (section 5.2.2 et 5.2.3). La section suivante (section 5.2.4) traite des objets que les agents aspectuels manipulent. Ensuite nous exposerons plus en détails ce qu'est un agent aspectuel et son fonctionnement (sections 5.3.1 et 5.3.2).

5.2.1 Éléments du système

Le système est composé de différents éléments :

- des agents ;
- des organisations ;
- des composants ;
- des messages et des objets.

Les agents sont dans des organisations qui elles mêmes sont regroupés dans des composants. L'ensemble des composants constitue le système entier (voir figure 5.2.1). L'intérêt de compartimenter les agents en organisations et elles même

en composants est de structurer le système, et de limiter les interactions entre certains types d'agents. Initialement, cela avait été proposé pour que chaque composant ait un module aspectuel, un module morphologique et un module d'analyse réifiés en trois différentes organisations.

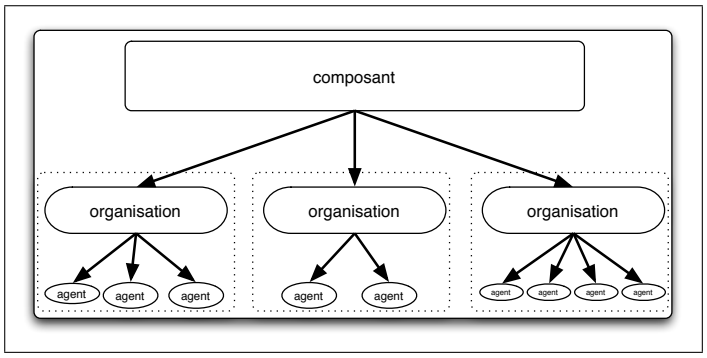


Figure 5.2.1 – *Schéma d'un composant*

Composants

Les composants sont les éléments de plus haut niveau dans le système. Ils gèrent les organisations. Ils sont responsables aussi pour la communication avec l'extérieur par le biais d'une interface. Tout message provenant de l'extérieur transite par le composant.

Organisations

Les organisations sont les éléments du système qui contiennent les agents. Elles sont responsables de la gestion des agents : création, ajout, clonage, retrait d'agent. Une organisation est responsable de la transition des messages pour les agents au sein d'elle même et en dehors d'elle. L'interaction avec une organisation se fait par l'envoi de messages à son adresse, par la biais d'une boîte aux lettres.

Agents

Les agents sont les éléments de base du système. Ce sont les agents qui « font le travail ». Les composants et les organisations n'ont véritablement qu'un rôle logistique et un rôle structurel.

5.2.2 Identification des éléments du système : les adresses

Système d'adressage

Les agents, les organisations et les composants possèdent tous un identifiant unique, appelé adresse. Cette adresse sert à l'identification des éléments du système pour le suivi, le déverminage, à l'envoi et la réception de message, à la communication entre les agents. L'adresse est une sorte de numéro de série. À chaque création d'un élément, une nouvelle adresse est créée.

La structure hiérarchique composant-organisation-agent est reflétée dans le système d'adressage. Une adresse est constituée de 3 parties, chacune spécifiant respectivement le composant, l'organisation et l'agent. Selon qu'une adresse identifie un agent, une organisation ou un composant, chacune de ces parties prend une valeur particulière. Il y a trois types de valeurs possibles :

- « aucun » ;
- « unique *n* » ;
- « tout ».

Ces valeurs désignent respectivement aucun élément, un élément et tout élément d'un certain niveau hiérarchique (soit composant, soit organisation ou soit agent). L'identifiant unique est accompagné d'un argument numérique *n* (un numéro de série) spécifiant un élément particulier. Cette structuration est inspirée des adresses IP : le « 255 » d'IP correspondrait à « tout », le « 0 » d'IP correspondrait à « aucun » et les valeurs entre « 1 » et « 254 » d'IP correspondraient aux valeurs « unique n ».

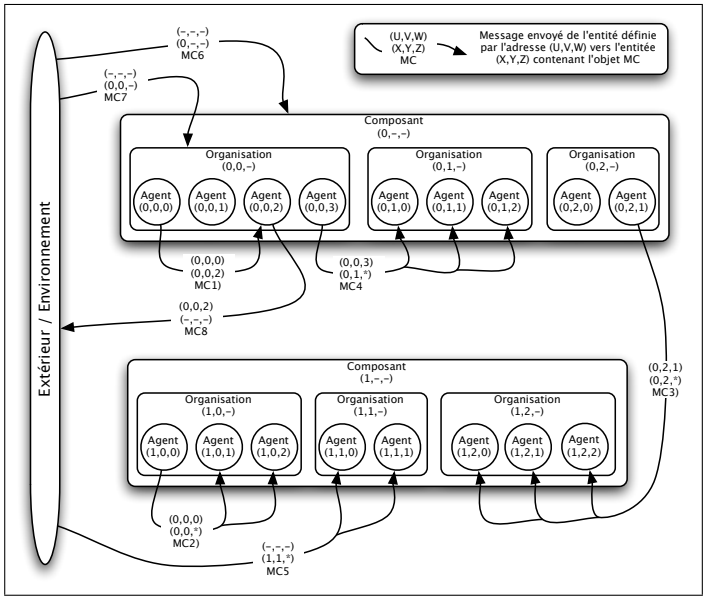


Figure 5.2.2 – *Exemple de système avec adresses des éléments.*

Adresses valides

Toutes les combinaisons de valeur-niveau ne sont pas autorisées. Les valeurs autorisées sont énumérées dans le tableau suivant :

Composant	Organisation	Agent	Élément(s) désigné(s) par l'adresse
Aucun	Aucun	Aucun	Adresse nulle
Aucun	Aucun	Unique n_a	N/A
Aucun	Aucun	Tout	N/A
Aucun	Unique n_o	Aucun	N/A
Aucun	Unique n_o	Unique n_a	N/A
Aucun	Unique n_o	Tout	N/A
Aucun	Tout	Aucun	N/A
Aucun	Tout	Unique n_a	N/A
Aucun	Tout	Tout	N/A
Unique n_c	Aucun	Aucun	Composant n_c
Unique n_c	Aucun	Unique n_a	N/A
Unique n_c	Aucun	Tout	N/A
Unique n_c	Unique n_o	Aucun	Organisation n_o du composant n_c
Unique n_c	Unique n_o	Unique n_a	Agent n_a de l'organisation n_o du composant n_c

Composant	Organisation	Agent	Élément(s) désigné(s) par l'adresse
Unique n_c	Unique n_o	Tout	Tout agent de l'organisation n_o du composant n_c
Unique n_c	Tout	Aucun	Tout organisation du composant n_c
Unique n_c	Tout	Unique n_a	Agent n_a de toute organisation du composant n_c
Unique n_c	Tout	Tout	Tout agent de toute organisation du composant n_c
Tout	Aucun	Aucun	Tout composant (sans ses éléments constitutifs)
Tout	Aucun	Unique n_a	N/A
Tout	Aucun	Tout	N/A
Tout	Unique n_o	Aucun	Organisation n_o de tout composant
Tout	Unique n_o	Unique n_a	Agent n_a de l'organisation n_o de tout composant
Tout	Unique n_o	Tout	Tout agent de l'organisation n_o de tout composant
Tout	Tout	Aucun	Toute organisation de tout composant
Tout	Tout	Unique n_a	Agent n_a de toute organisation de tout composant
Tout	Tout	Tout	Tout agent de toute organisation de tout composant

5.2.3 Communication entre les éléments : les messages

Structure d'un message

Les agents communiquent entre eux par le biais de messages. Ce sont des messages réifiés, et non de simples appels d'une méthode sur un objet comme dans la programmation orientée objet.

Un message est constitué des 3 parties :

- la source ;
- la destination ;
- le contenu du message.

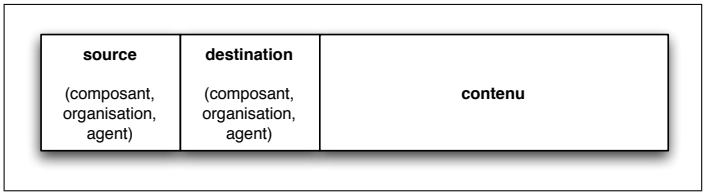


Figure 5.2.3 – *Un message est constitué d'un triplet (source, destination, contenu)*

La source, qui désigne l'expéditeur, est utile pour le destinataire, pour continuer éventuellement les échanges, ou mettre à jour certaines informations en relation avec la source, notamment s'il s'agit d'agents. La destination désigne le ou les destinataires du message.

Types de messages

On peut distinguer deux grands types de messages, selon leur contenus, qui transitent entre les éléments du système :

- les messages de fonctionnement du système ;
- les messages relatifs à la sémantique du problème traité par le système.

Les messages du premier type touchent à tout ce qui concerne le fonctionnement proprement dit du système. Cela regroupe les messages de demande de création de groupes, de fusion de groupes, d'échange d'« énergie »... que nous verrons par la suite.

Les messages de second type représentent les objets manipulés par les agents. Ces messages sont dépendants du problème traité par le système, de l'ontologie établie. Par exemple, si le système doit traiter des images, les objets pourront décrire les concepts liés aux formes, au couleurs...

Envois valides de messages

Un élément ne peut pas envoyer un message à n'importe quelle adresse. Voici les combinaisons de (source, destination) valides :

			Adresse possible pour	
Composant	Organisation	Agent	source	destination
Aucun	Aucun	Aucun	Oui	Non
Aucun	Aucun	Unique n_a	N/A	N/A
Aucun	Aucun	Tout	N/A	N/A
Aucun	Unique n_o	Aucun	N/A	N/A
Aucun	Unique n_o	Unique n_a	N/A	N/A
Aucun	Unique n_o	Tout	N/A	N/A

			Adresse possible pour	
Composant	Organisation	Agent	source	destination
Aucun	Tout	Aucun	N/A	N/A
Aucun	Tout	Unique n_a	N/A	N/A
Aucun	Tout	Tout	N/A	N/A
Unique n_c	Aucun	Aucun	Oui	Oui
Unique n_c	Aucun	Unique n_a	N/A	N/A
Unique n_c	Aucun	Tout	N/A	N/A
Unique n_c	Unique n_o	Aucun	Oui	Oui
Unique n_c	Unique n_o	Unique n_a	Oui	Oui
Unique n_c	Unique n_o	Tout	Non	Oui
Unique n_c	Tout	Aucun	Non	Oui
Unique n_c	Tout	Unique n_a	Non	Oui
Unique n_c	Tout	Tout	Non	Oui
Tout	Aucun	Aucun	Non	Oui
Tout	Aucun	Unique n_a	N/A	N/A
Tout	Aucun	Tout	N/A	N/A
Tout	Unique n_o	Aucun	Non	Oui
Tout	Unique n_o	Unique n_a	Non	Oui
Tout	Unique n_o	Tout	Non	Oui
Tout	Tout	Aucun	Non	Oui
Tout	Tout	Unique n_a	Non	Oui
Tout	Tout	Tout	Non	Oui

Les adresses de type « un agent de toute organisation de tout composant » sera relativement peu utilisée dans les émissions de messages. L'adresse « nulle », ne désignant pas d'élément en particulier, est utilisée pour des messages provenant de l'extérieur, de l'interface, ou bien dans des situations particulières, notamment des tâches « administratives ».

Chaque élément du système possède une « boîte aux lettres » pour l'envoi et la réception de messages. L'identifiant de la boîte aux lettres est celle de l'élément en question. Si un élément envoie un message à une adresse invalide ou obsolète (dans le cas ou un agent est retiré du système par exemple), celui-ci aura une réponse par un message notifiant que l'adresse est invalide.

5.2.4 Objets

Les objets mentionnés ici ne font pas directement référence à la notion d'objet de la programmation orientée objet.

Description des objets

Les objets sont sur quoi les agents aspectuels opèrent. Les objets sont passifs. Ils sont de l'information que les agents aspectuels lisent, traitent et produisent. Si le système est couplé à l'extérieur par une interface, les informations qui intéressent le système seront encodées sous forme d'objets (définis par l'ontologie préalablement constituée). Les objets transitent dans le système en étant encapsulés dans un message. Un agent ne peut lire les informations contenues dans un objet que si l'objet est contenu dans un message qui est destiné à l'agent. Ceci est notamment le cas pour les messages destinés à un ensemble d'agents ou à un agent en particulier.

Lorsqu'un agent produit des objets et souhaite les diffuser à d'autres agents, il encapsule l'objet dans un message dont la source est son adresse. L'agent peut envoyer son message à un ou plusieurs éléments du système désignés par des adresses valides.

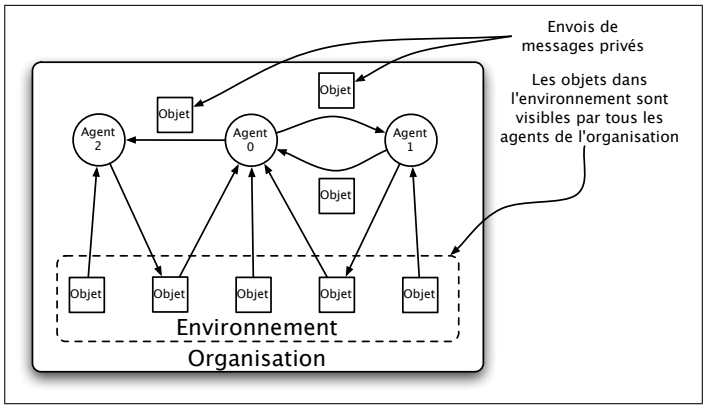


Figure 5.2.4 – *Les objets sont disponibles pour les agents dans l'environnement de l'organisation. Ou bien les objets transitent d'un agent à un autre dans un message.*

Les objets sont définis par une étiquette (pour la sémantique) éventuellement complétée par une liste de paramètres caractérisant l'objet. Les paramètres peuvent eux-mêmes être des objets. Voici quelques exemples d'objets :

- une figure géométrique : un cercle de diamètre r, une ellipse de rayons a et b, un carré de longueur $l\ldots$;
- une quantité numérique arbitraire X ;
- une couleur définie par un triplet (r,g,b) ;
- $\ldots$

L'ensemble des objets est défini dans ce qu'on appelle l'espace des objets, noté E. On peut faire l'analogie avec un espace vectoriel ou une variété, où chaque objet défini est un point de cet espace, chaque dimension de l'espace étant alors une caractéristique des objets. Afin de permettre aux agents de comparer les objets et de les manipuler, l'espace des objets sera muni d'une fonction d'agrégation et d'une métrique (ou pseudo-métrique). Remarquons que les définitions n'auront quand même pas la rigueur de celles utilisées en Mathématiques.

Composition d'objets

Certains objets peuvent se composer pour définir de nouveaux objets. Par exemple, on définit les concepts suivants :

- forme (cercle, ellipse, carré, rectangle, triangle, $\ldots$) ;
- position (couple (x,y)) ;
- couleur (triplet (r,g,b)).

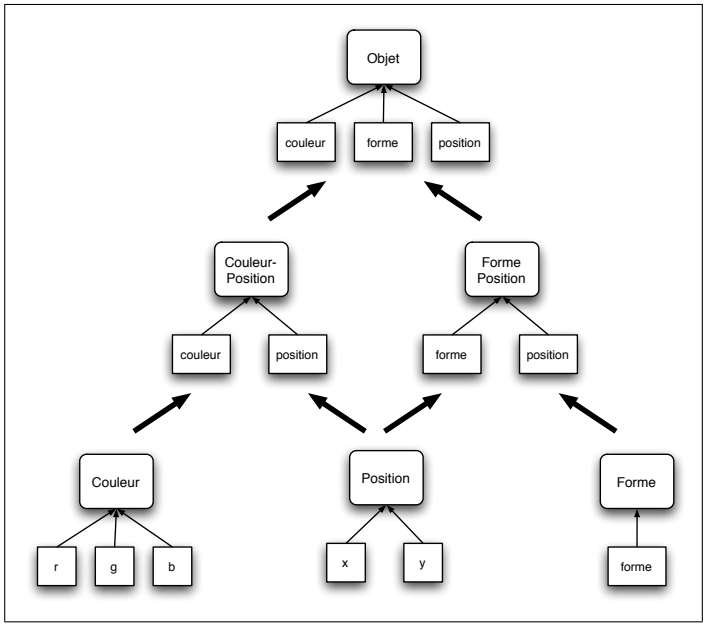


Figure 5.2.5 – *Schéma d'un exemple d'ontologie d'objets et de sa hiérarchie.*

Il est possible de définir un quatrième concept, « un objet », au sens « objet physique » et non objet manipulé par les agents, à partir des concepts précédents, par la composition (voir figure 5.2.5) :

- d'une forme donnée ;
- d'une position donnée ;
- d'une couleur donnée.

Ce nouveau concept d'objet physique, permettra de réifier le fait que si l'on observe une couleur à une position donnée et une forme à la même position, ou à peu près, c'est qu'il y a de fortes chances d'avoir un objet de cette forme et de cette couleur à cette position là. Ceci a été inspiré du mécanisme de traitement de certains types d'informations dans le cerveau. Par exemple, pour les informations visuelles, la couleur « emprunte un chemin » différent de celui des formes.

Fonctions d'agrégation d'objets et listes d'objets

Agrégation d'objets L'espace E des objets manipulés par les agents est doté d'une fonction d'agrégation. À la différence de la composition, on pourrait dire que l'agrégation produit un objet de même niveau conceptuel. Par exemple, l'agrégation de deux objets désignant des positions p_1 et p_2 peut être définie comme le point équidistant de p_1 et de p_2. Plus généralement, on peut définir l'agrégation comme le barycentre d'objets : $f_{agr} : E \times \mathbb{R} \times E \times \mathbb{R} \rightarrow E \times \mathbb{R}$, où f_{agr} : $(o_1, c_1), (o_2, c2) \mapsto (o, c)$. Pour des objets de même classe, structurellement définis de la même façon, il est facile de définir cette fonction. Pour des objets qui sont sémantiquement proches, mais définis de façon différente, on peut essayer de définir une fonction. Par exemple, l'agrégation d'un cercle $C(r)$ de rayon r et une ellipse $E(a,b)$ de rayon a et b pourrait donner une ellipse $E(\frac{r+a}{2}, \frac{r+b}{2})$. L'agrégation peut aussi être pondérée par des coefficients, c_1 pour le cercle et c_2 pour l'ellipse, ce qui donnerait : $E(\frac{c_1 r + c_2 a}{c_1 + c_2}, \frac{c_1 r + c_2 b}{c_1 + c_2})$.

La définition d'une fonction d'agrégation de deux objets reste cependant arbitraire, c'est un choix dépendant des ontologies utilisées.

Liste d'objets : agrégation par défaut Si pour un couple de classes d'objets, il n'y a pas de fonction d'agrégation définie, on les regroupe sous forme de liste d'objets, tout simplement. Par exemple, l'agrégation d'une couleur c et d'une position p n'a, a priori, pas de sens (du moins commun), et sera définie comme $[c; p]$. On peut cependant noter que rien ne nous empêche de définir une classe d'objet « afv3GRZ » qui peut se définir à partir d'une couleur et d'une position. C'est à l'utilisateur de définir ces fonctions.

Agrégation d'un objet et d'une liste d'objets L'objet est agrégé à un objet de la liste, et la liste est agrégée jusqu'à l'atteinte d'un point fixe : l'agrégation de la liste donne la liste elle-même.

Agrégation de deux listes d'objets De façon similaire, l'agrégation de deux listes consiste à agréger chaque objet d'une liste à ceux de l'autre liste. Plus simplement, on concatène les deux listes et l'agrégation opère sur cette liste d'objets.

Difficultés de la réalisation de l'agrégation Il n'est pas toujours évident de définir une fonction d'agrégation pour n'importe quelle paire d'objets et encore moins de trouver une fonction qui satisfasse à des critères d'unicité ou d'invariance à l'ordre de l'agrégation.

Métrique sur l'espace d'objets

Une autre fonction est introduite dans l'espace des objets. Il s'agit de la fonction de calcul de « distance » (voir la figure 5.2.6). La définition de cette fonction n'aura pas forcément la rigueur de la distance mathématique (notamment en ce qui concerne la relation de Chasles, inégalité triangulaire), $f_{dist} : E \times E \rightarrow \mathbb{R}$

On définit, de façon similaire les fonctions sur des ensembles : $\hat{f}_{dist} : P(E) \times P(E) \rightarrow \mathbb{R}$, qui prend deux parties de E, deux listes d'objets, et renvoie la distance entre ces deux ensembles.

La distance entre un objet o et une liste d'objets l peut être définie, par exemple, comme :

- soit la somme des distances entre l'objet o et chaque objet de la liste l ;
- soit le maximum des distances entre l'objet o et chaque objet de la liste l ;
- soit la moyenne des distances entre l'objet o et chaque objet de la liste l ;
- ...

L'intérêt de doter l'espace des objets d'une telle fonction est de donner aux agents un moyen de comparer des objets (voir figure 5.2.6).

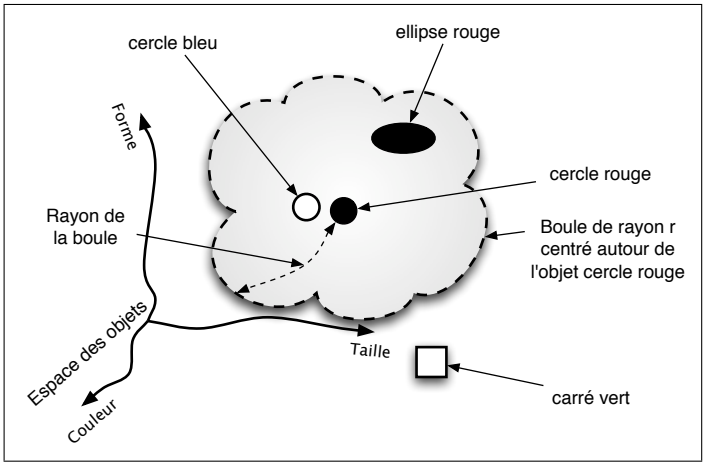


Figure 5.2.6 – *Schéma d'un exemple d'espace des objets avec les dimensions de couleur, forme, taille. Une boule d'un certain rayon est centrée autour de l'objet « cercle rouge » ; le « cercle bleu » et l'« ellipse rouge » sont dans le voisinage du « cercle rouge » défini par la boule en question. En revanche, le « carré vert », plus loin, est situé à l'extérieur de la boule.*

Encore une fois, c'est à l'utilisateur que revient la tâche de définir les distances entre objets, selon ses choix.

5.2.5 Agents aspectuels

Les agents aspectuels sont les éléments minimaux qui traitent l'information relative au problème étudié. Sans pour autant prétendre que le système peut traiter tout type de problème, notre système a été construit de telle sorte que les agents aient un comportement générique indépendant de la sémantique du problème. Le comportement de l'agent, sa fonction au sein du système, est spécifié en lui attribuant un rôle particulier. Le fonctionnement de l'organisation aspectuelle étant relativement complexe, cette section en donne une première description générale, introduisant les concepts importants et leurs relations. Une description plus détaillée sera donnée dans la section 5.3.

Notion de rôle de l'agent aspectuel

Un agent aspectuel simple a un comportement générique paramétrable. Cependant la fonction de l'agent, défini par son rôle, pilote ce comportement. Il y a deux dimensions de variabilité du comportement d'un agent. Dans un même contexte donné, deux agents pourront agir de façon différentes parce que :

- ils ont des paramètres différents (valeurs de seuils d'activation, énergie, historique...) ;
- qu'ils ont des rôles différents.

Concrètement le rôle de l'agent détermine le contexte dans lequel il peut s'activer, ce qu'il fait, et éventuellement quels objets il produit. On définit le rôle comme un triplet $R(r_{nom}, f_{activation}, f_{production})$. Le paramètre r_{nom} est la description du rôle, la fonction $f_{activation} : P(E) \rightarrow \mathbb{R}$ sert à déterminer la pertinence du rôle dans un contexte donné, la fonction de $f_{production} : P(E) \rightarrow P(E)$ génère une liste d'objets variant selon le contexte.

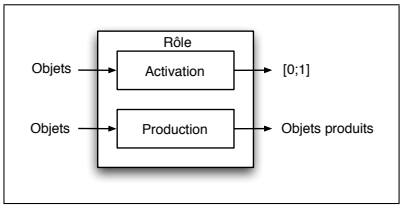


Figure 5.2.7 – *Schéma d'un rôle.*

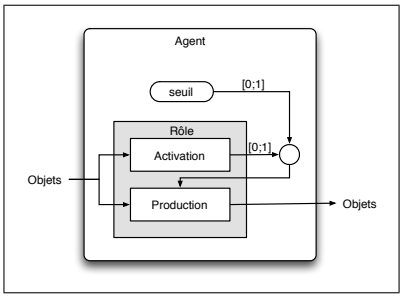


Figure 5.2.8 – *Utilisation du rôle dans l'agent.*

Par exemple, un agent dont le rôle est de détecter une forme particulière d'une couleur donnée s'activera s'il perçoit des objets décrivant une forme f à une position p_1 et des objets décrivant une couleur c à une position p_2 si p_1 et p_2 sont « suffisamment » proches. Il produira alors un objet spécifiant une forme colorée $o(f,c,p)$.

Interactions entre les agents aspectuels

Les agents aspectuels peuvent apprécier les autres agents aspectuels selon les objets qu'ils produisent et qu'ils mettent à disposition dans l'environnement. Deux agents qui produisent des objets, ou des listes d'objets, proches en terme de distance auront tendance à se considérer comme alliés potentiels et auront de forte chance de s'« entraider », en revanche des agents produisant des objets sémantiquement distants seront perçus mutuellement comme des ennemis possibles et entreront en général en conflit. Les agents peuvent connaître le rôle des autres agents, mais ils ne se servent pas directement de cette connaissance. La façon normale pour un agent d'estimer la position d'un autre agent est par l'analyse des objets que ce dernier produit par rapport à sa propre production. Nous avons retenu ce mode de comportement, car deux agents de même rôle mais avec des historiques différents, pourraient produire différents objets. Néanmoins nous verrons ultérieurement que l'agent (notamment l'agent groupe) utilise cette connaissance du rôle des autres agents pour opérer une restructuration de l'organisation, en réponse à une injonction de la morphologie (par exemple lors de la reconfiguration des groupes d'agents).

Chaque agent possède un niveau dit d'« énergie ». On peut voir cette énergie comme un potentiel d'action. Plus un agent a un niveau d'énergie élevé, plus il aura de marge de manœuvre. En revanche un agent ayant une énergie nulle tombera en état de veille et ne pourra effectuer quoi que ce soit pendant un certain temps. En temps normal l'agent doit faire en sorte que son rôle et les objets qu'il produit soient significatifs, si le contexte le permet, et doit réduire les agents qui proposent la génération d'objets éloignés, ou aider ceux qui suggèrent des objets proches. Cependant ce comportement, dit « normal », peut être modifié en révisant les règles contenues dans son module de comportement, décrit plus loin. À cette fin, les agents peuvent renforcer ou réduire d'autres agents en envoyant des messages appropriés. Pour réduire un agent, un agent envoie un message d'« attaque » à cet agent. Pour l'aider directement, il envoie un message d'« énergie ». Les agents peuvent s'aider indirectement en formant des groupes.

Agents aspectuels de type groupe

Les agents aspectuels simples ont la possibilité, à leur initiative, de former un groupe afin de s'associer pour mutualiser leur énergie en vue de mieux résister aux réductions provenant d'autres agents aspectuels et imposer leurs positions plus fortement. Cette notion de groupe est réifiée par un type d'agent spécifique : l'agent groupe (voir figure 5.2.9).

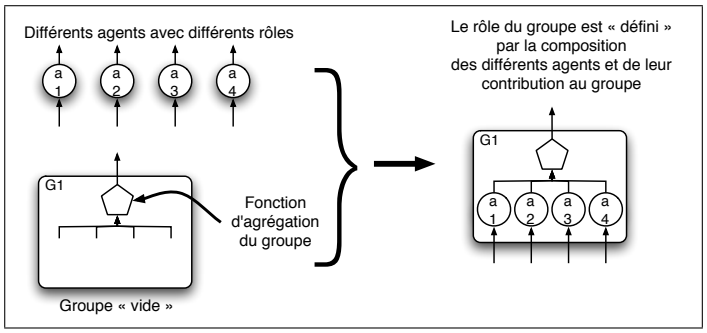


Figure 5.2.9 – *Schéma d'un agent groupe. Un groupe est constitué de plusieurs agents simples. Un groupe est réifié par un agent de type groupe. Le rôle de l'agent groupe est spécifié implicitement par le rôle des agents simples le constituant. La production des objets de l'agent groupe est issue de l'agrégation des objets de ses agents simples.*

Notion de rôle d'un groupe

Un agent groupe peut être vu comme un agent dont le rôle est implicite et évolue dans le temps de façon dynamique. En effet, le groupe n'a pas de rôle explicitement défini. Le rôle du groupe est défini implicitement par l'activité des agents aspectuels qui le composent. Le comportement de l'agent groupe et la notion de son rôle implicite sont décrits plus loin.

5.3 Comportement des agents simples et des agents groupes

Cette section présente avec davantage de précision le comportement des agents aspectuels.

5.3.1 Agent simple

Nous allons préciser la structure de l'agent aspectuel maintenant que nous avons une idée générale du comportement de l'agent « simple », par opposition à l'agent « groupe ». Voici les éléments principaux caractéristiques de l'agent aspectuel simple :

- un état : un agent aspectuel simple a trois états généraux :
  - l'état de veille,
  - l'état d'activité
  - l'état de « mort » ;
- un évaluateur de contexte : cela permet de résumer symboliquement son état et les interactions de l'agent avec les autres agents ;
- un module de règle de comportement : ce module définit les actions à effectuer selon le contexte évalué ;
- un rôle : il définit ce que l'agent fait dans l'organisation en tenant compte de la sémantique du problème. Ceci est spécifié par l'utilisateur ;
- un carnet d'adresses : les préférences d'agents. Initialement vierge, il reflète l'historique des interactions avec les autres agents ;
- un groupe (éventuellement) : l'agent peut ou non appartenir à un groupe, auquel il contribue.

Le comportement générique de l'agent est résumé sur la figure 5.3.1. Lorsqu'un agent est créé, il démarre à « Éveil ». Ensuite il entre dans la boucle générale. L'agent consulte les messages contenus dans sa boîte aux lettres. Il réagit de façon automatique à certains types de messages, comme l'apport d'énergie ou des messages de logistique. Ensuite, selon son état, actif, en veille ou mort, l'agent agit de façon différente. Si l'agent est mort, il est retiré du système. S'il est endormi, il attend un certain nombre de cycles avant de pouvoir s'éveiller à nouveau. S'il est actif, l'agent entre dans la partie « autonome » de son comportement qui peut

être modulé par son histoire. Le module de comportement, son carnet d'adresses, reflète son expérience passée et influence les actions que l'agent entreprend.

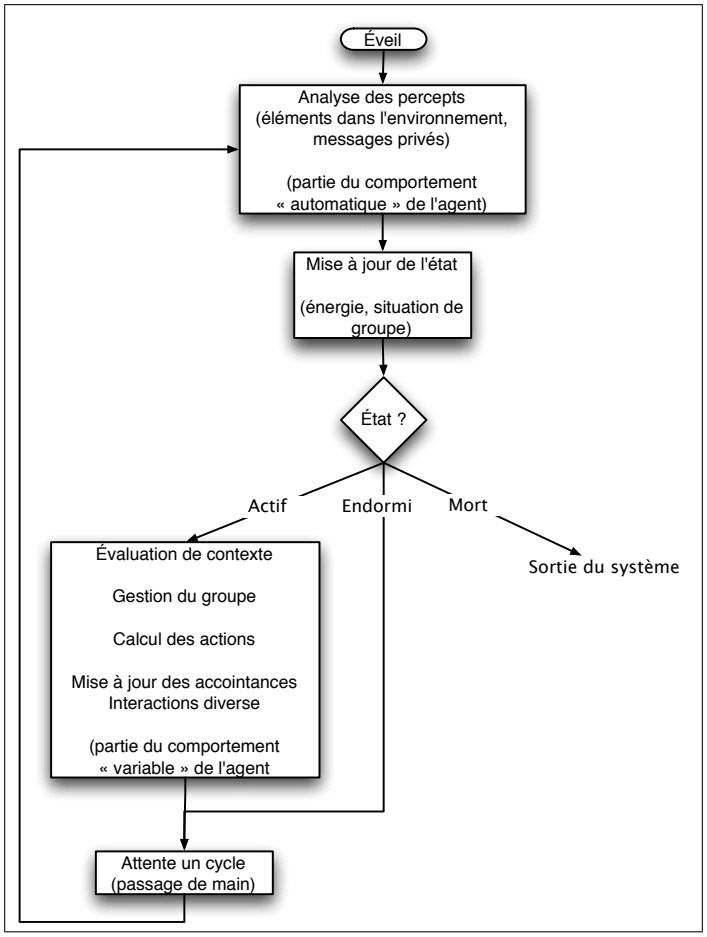


Figure 5.3.1 – *Organigramme général du comportement d'un agent aspectuel*

État de veille de l'agent aspectuel simple

L'état de veille n'est pas très productif pour l'activité du système, mais l'agent peut s'y trouver de temps à autre. Un agent aspectuel tombe dans cet état lorsqu'il n'a plus d'énergie. Cela peut arriver si l'agent est trop souvent attaqué, le rythme des réductions qu'il subit est plus élevé que celui par lequel il parvient à récolter de l'énergie, soit en puisant gracieusement dans l'environnement, soit par la générosité d'autres agents ou bien encore par le système.

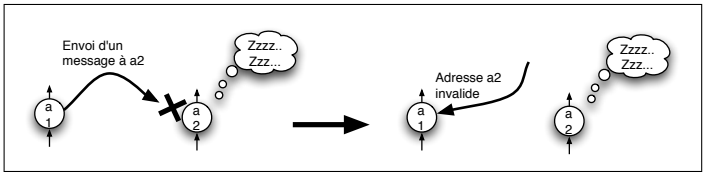


Figure 5.3.2 – *L'agent a*1 *tente d'envoyer un message à l'agent a*2 *qui est en veille. L'agent a*1 *reçoit un message indiquant que l'adresse a*2 *est momentanément invalide.*

C'est un état dans lequel l'agent ne fait presque rien. Il ne peut émettre de message ni recevoir de message de la part d'autre agents. Les agents qui envoient un message vers un agent en veille ont un message de retour de la part du système en disant que l'adresse est invalide (voir figure 5.3.2). C'est à dire qu'aucun message ne peut y parvenir pour l'instant. Un agent dans cet état n'évalue même pas son contexte.

Un agent aspectuel en veille finira éventuellement par s'éveiller au bout d'un certain temps, son niveau d'énergie est alors réinitialisé à un niveau positif.

État de mort d'un agent

L'agent ne fait plus rien dans cet état. Un agent qui se trouve dans cet état sera retiré du système.

État d'activation de l'agent aspectuel simple

Cet état est celui où l'agent est en action. C'est l'état dans lequel il se trouve initialement au lancement du système, à moins qu'une configuration particulière du système ne soit adoptée.

Évaluation du contexte de l'agent aspectuel simple

Un agent activé peut évaluer son contexte. L'évaluation du contexte passe par le calcul de certaines propriétés, par exemple :

- si la fonction d'activation de son rôle dépasse un seuil d'activation ;
- si l'agent se trouve dans un groupe ;
- s'il est attaqué ;
- s'il a reçu de l'énergie ;
- si son niveau d'énergie est élevé ou bas ;
- s'il a perdu ou gagné de l'énergie.

Ceci est effectué au début du cycle d'exécution de l'agent (voir figure 5.3.1). Pour tester l'activation de son rôle, l'agent aspectuel activé explore son environnement et consulte sa boîte aux lettres.

Au niveau de l'implantation, l'évaluation du contexte s'effectue par l'attribution d'une valeur en 0 et 1 à un symbole donné spécifiant la sémantique du contexte (par exemple `(ct_low_enegy, 0.9)`, `(ct_in_group, 0.0)`...).

Bilan énergétique

La consultation de la boîte aux lettres peut révéler des messages contenant de l'énergie ou des attaques. L'état de l'agent est mis à jour en conséquence : les messages d'attaque lui font perdre de l'énergie ; les messages d'énergie font augmenter son niveau d'énergie. Rappelons que lorsque le niveau d'énergie devient négatif, l'agent se met en veille.

La mise à jour du niveau d'énergie se fait séquentiellement, si bien que si son niveau d'énergie baisse en dessous de zéro alors qu'il lui reste encore des messages d'apport d'énergie, il bascule tout de même en léthargie, et les messages d'énergie sont à jamais perdus. Ceci est un choix d'implantation, il aurait été possible de faire un bilan énergétique avant de mettre à jour l'état de l'agent.

Les différents événements sont mémorisés dans le contexte, par exemple, s'il s'est fait attaquer ou s'il a gagné de l'énergie ainsi que le bilan énergétique.

Si un agent est attaqué par un autre agent il revoit à la baisse sa préférence pour cet agent agresseur. En revanche, s'il reçoit de l'énergie d'un agent, il augmente sa préférence pour cet agent. S'il ne s'agit pas d'un agent (par exemple il puise de l'énergie dans l'environnement), il ne fait rien de particulier.

Activation du rôle

L'agent extrait de l'environnement l'ensemble des objets. Cette liste d'objets est passée en argument à la fonction d'activation de son rôle. Celle-ci retourne une valeur entre 0 et 1. Cette valeur est comparée avec le seuil d'activation de l'agent. Le résultat de cette comparaison est stocké dans le contexte de l'agent. Ce résultat servira ultérieurement dans le calcul des règles comportementales à déclencher.

Mise à jour des préférences

Les préférences spécifient les affinités d'un agent pour les autres agents. Selon les paramètres de l'agent, celui-ci aura tendance à « aider », « coopérer » avec les agents dont ses préférences sont positives, et inversement si ses préférences sont négatives. Les préférences d'un agent peuvent :

- être initialisées à la création d'un agent ;
- évoluer dans le temps en fonction des interactions entre agents ;
- être modifiées par des injonctions spécifiques émanant de l'analyse.

En fonction des objets qu'il a pu percevoir dans l'environnement, l'agent met à jour ses préférences. La réévaluation d'une préférence d'un agent pour un autre est fonction de :

- l'appréciation d'un agent par un autre agent. L'appréciation dépend de la distance entre les objets produits par chacun des agents et du rayon de l'agent. La distance entre les deux ensembles d'objets est comparée au rayon de l'agent ;
- des interactions entre les deux agents.

L'évaluation de la préférence en fonction des objets produits peut être asymétrique, si les agents ont des rayons différents. Par exemple, dans le cas présenté

dans la figure 5.3.3 :

- du point de vue de l'agent $a1$, l'agent $a2$ est vu comme un allié. Ainsi l'agent $a1$ augmentera sa préférence pour l'agent $a2$;
- inversement, l'agent $a2$ perçoit l'agent $a1$ comme un ennemi, car ce dernier est en dehors de sa boule, et baissera sa préférence pour l'agent $a1$.

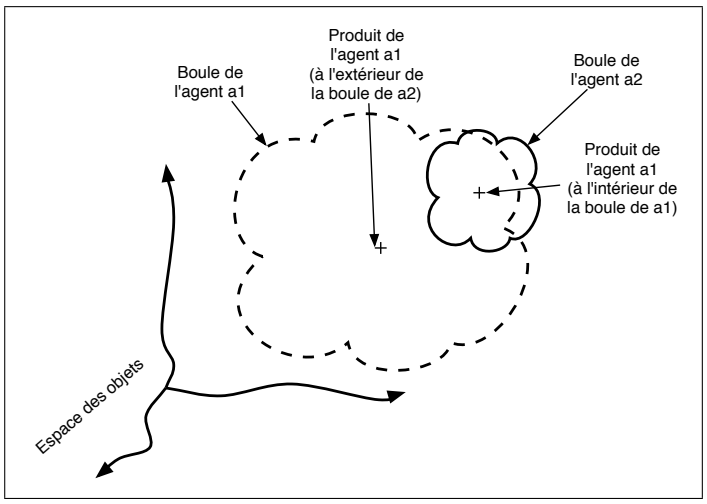


Figure 5.3.3 – *Préférence asymétrique. Le rayon de l'agent $a1$ étant plus grand que celui de l'agent $a2$, l'agent $a1$ considère $a2$ comme un allié potentiel alors que ce n'est pas le cas pour l'agent $a2$.*

Les préférences interviennent dans le comportement de l'agent et dans les choix des agents avec lesquels il va s'associer, ou bien avec ceux qu'il va combattre et tenter de réduire. Les agents évaluent bien entendu les agents simples, mais également les agents groupes qui, de l'extérieur, s'apparentent à des agents simples.

Activation des règles de comportement

Le module de comportement de l'agent aspectuel dicte le comportement de l'agent. Ce module consiste en fait en un ensemble de règles. Ces règles ont la forme commune « Si ... alors ... ». Voici une liste non exhaustive des symboles de contexte :

- `Ct_in_group` : l'agent est-il dans un groupe ? ;
- `Ct_percepts_init` : l'environnement de l'agent est-il favorable à son activation ? ;
- `Ct_receive_energy` : l'agent a-t-il reçu de l'énergie ? ;
- `Ct_attacked` : l'agent a-t-il été attaqué ? ;
- `Ct_products_sent` : l'agent a-t-il émis des objets ? ;
- `Ct_energy_high` : le niveau d'énergie de l'agent est-il élevé ? ;
- `Ct_energy_low` : le niveau d'énergie de l'agent est-il bas ? ;
- `Ct_energy_loss` : l'agent a-t-il gagné de l'énergie ? ;
- `Ct_energy_gain` : l'agent a-t-il perdu de l'énergie ? ;
- `Ct_own_group_good` : l'agent est-il satisfait de son groupe ? ;
- ...

Et un extrait de la liste d'actions qui peuvent être entreprises par l'agent par le biais de son module de comportement :

- `Act_donate_energy_group` : donner de l'énergie au groupe ;
- `Act_send_symbols` : générer des produits ;
- `Act_be_altruist` : devenir davantage « altruiste » ;
- `Act_join_group` : rejoindre un groupe ;
- `Act_leave_group` : quitter un groupe ;
- `Act_attack` : attaquer un agent ;
- `Act_less_picky` : devenir moins exigeant ;
- `Act_more_picky` : devenir plus exigeant ;
- ...

Voici quelques exemples de règles utilisées dans les expériences. Une règle est composée de deux listes, la première comporte la condition sous forme de conjonction, et la seconde correspond à la liste d'actions à effectuer en cas d'activation de la règle :

```
[ Ct_in_group,              0.0 ;  Ct_percepts_init,        1.0 ;]
[ Act_join_group,           1.0 ;]
```

Ceci peut se lire de la façon suivante : « Si je ne suis pas dans un groupe et que mon environnement m'est favorable, alors rejoindre un groupe ».

```
[ Ct_own_group_good,        0.0 ;]
```

```
[ Act_leave_group,          1.0 ;]
```

Ceci signifie : « Si le groupe dans lequel je me trouve ne me satisfait pas, alors je quitte le groupe »

```
[ Ct_energy_high,           1.0 ;  Ct_in_group,             1.0 ]
[ Act_donate_energy_group, 1.0 ]
```

Ceci peut être interprété comme : « Si mon niveau d'énergie est élevé et que je suis dans un groupe, alors donner de l'énergie à mon groupe ».

```
[ Ct_energy_high,           1.0 ;  Ct_in_group,             1.0 ]
[ Act_attack,               1.0 ]
```

Ceci peut être lu comme : « Si mon niveau d'énergie est élevé et que je suis dans un groupe, alors attaquer un agent adverse ».

Lorsque toutes les règles sont évaluées, la moyenne de leurs réponses est calculée pour déterminer quelles actions sont véritablement effectuées. Seules les actions dont le coefficient associé dépasse une valeur seuil, par exemple posée à 0.5 sur une échelle de $[0;1]$, seront effectuées. La valeur des coefficients, et les règles elles-mêmes, peuvent être modifiées au cours du temps.

5.3.2 Agent aspectuel groupe

L'agent groupe permet aux agents aspectuels simples de former des coalitions explicites pour bénéficier de l'effet de nombre et ainsi mieux résister aux tentatives de réductions des autres agents. Cependant, la contrepartie de cela est que les produits proposés par une agent simple dans le groupes sont agrégés avec les produits des autres agents du groupe, au risque de voir ses produits significativement peu représentés.

Un groupe est spécifié par :

- son état de fusion (seul, en cours de fusion, fusionné) ;
- son état de fusion cible (seul, fusionné) ;
- la liste d'agents aspectuels simples le composant ;
- son rayon ;
- son énergie ;
- l'historique des contributions des agents simples le constituant ;
- le rôle cible, éventuellement, s'il a reçu un ordre de la morphologie de modifier son rôle (voir plus loin la partie sur le contrôle morphologique).

Rôle agrégateur de l'agent groupe

Les agents aspectuels simples contribuent au groupe en lui fournissant de l'énergie. Il forment en quelque sorte un « super » agent piloté par l'agent groupe.

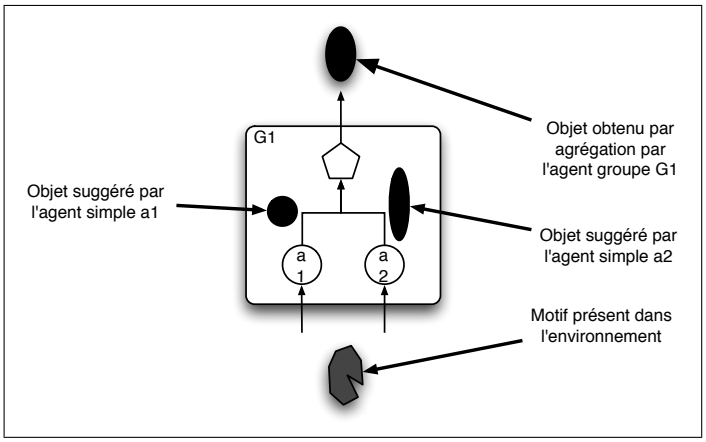


Figure 5.3.4 – *Agrégation d'agents simples dans un agent groupe.*

Si les agents aspectuels simples qui constituent le groupe produisent des objets, ils ne peuvent plus les émettre directement mais envoient leurs objets à leur agent groupe. Celui-ci agrège tous les objets qui lui sont envoyés par ses agents constituants au prorata des quantités d'énergie perçues. L'avantage de bénéficier de la protection mutuelle du groupe est contre-balancée par les faits de voir son rôle éventuellement moins bien représenté par le groupe.

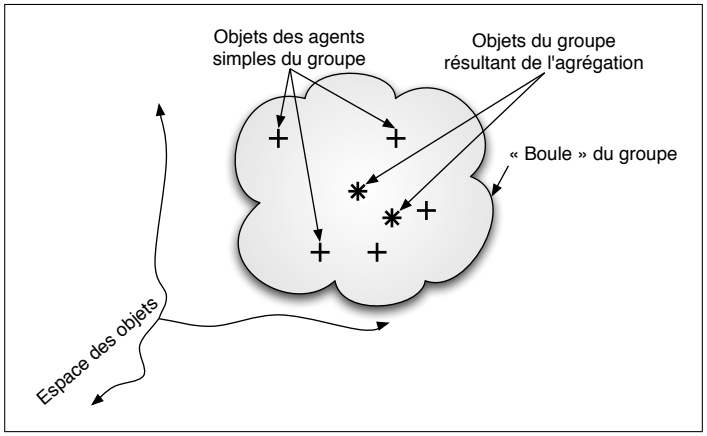


Figure 5.3.5 – *Représentation schématique des produits du groupe, issues des objets de ses agents constitutifs.*

Cycle de vie d'un groupe

Un agent aspectuel a la possibilité de créer un groupe, de se joindre à un groupe et de quitter un groupe. L'intégration d'un agent simple dans un groupe et le départ de l'agent simple d'un groupe ne sont pas automatiques.

Un groupe peut contenir de un à tous les agents d'une organisation. Un groupe qui ne contient plus d'agent disparaît. Seuls les agents d'une même organisation peuvent faire partie d'un même groupe.

Création d'un groupe À tout moment un agent simple peut faire la demande de création d'un groupe. Les seules restrictions sont lorsqu'il est mort, en veille ou s'il est déjà dans un groupe. Dans ce dernier cas, il doit d'abord quitter le groupe pour en former un autre. Par ailleurs un agent ne peut créer qu'un seul groupe à la fois (il ne peut être que dans un seul groupe à la fois). Techniquement la création d'un groupe passe par la diffusion d'un message, qui est intercepté par un agent particulier, un agent générateur d'agent groupe.

Intégration dans un groupe existant Lorsqu'un agent souhaite intégrer un groupe, il émet une demande d'intégration au groupe (figure 5.3.6).

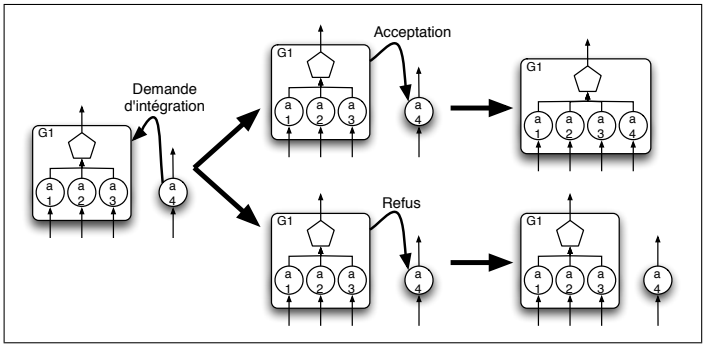


Figure 5.3.6 – *Intégration d'un agent dans un groupe. Le groupe peut refuser ou accepter l'agent en son sein, en fonction des objets de l'agent simple, ceux du groupe, du rayon du groupe.*

Cette demande contient la liste d'objets produits par l'agent qui fait la demande. Le groupe reçoit la demande. Le groupe estime la distance de la position de l'agent en comparant sa liste actuelle d'objets et celle contenue dans la demande. Le groupe peut accepter ou refuser l'agent en son sein :

- le groupe accepte l'agent, si la distance entre les deux ensembles d'objets est inférieure au rayon du groupe (figure 5.3.7) ;
- le groupe rejette la demande de l'agent (figure 5.3.8), lorsque celui-ci se trouve en dehors de la boule du groupe.

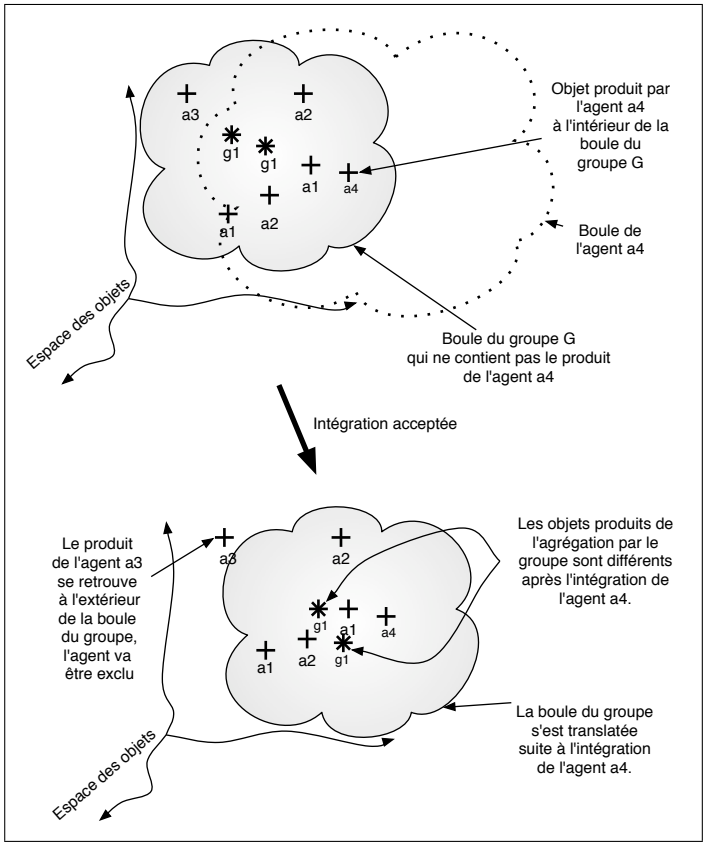


Figure 5.3.7 – *Le groupe accepte l'agent a4 car ses produits sont dans la boule du groupe. À l'issue de l'intégration de ce nouvel agent, les produits du groupe se translatent. Il se peut qu'un des agents du groupe (ici a3) sorte alors de la boule, cet agent sera éjecté du groupe.*

L'intégration de nouveaux agents simples dans le groupe peut avoir comme conséquence de modifier le rôle implicite de l'agent groupe. Par suite, il se peut que certains agents simples du groupe se retrouvent en dehors de la boule de l'agent groupe, ou bien que l'agent groupe sorte de la boule de certain de ses propres agents (voir figure 5.3.7).

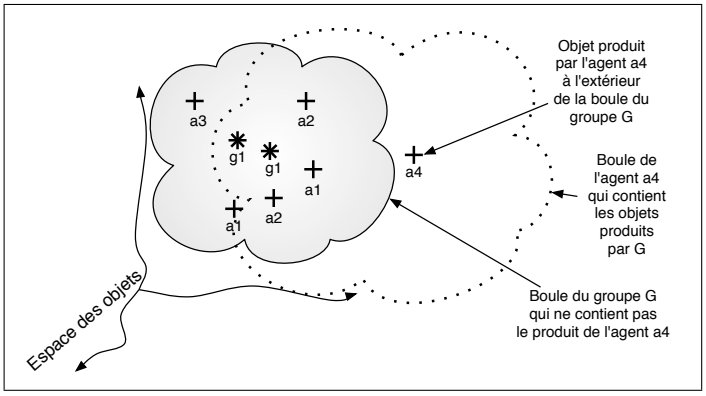


Figure 5.3.8 – *L'agent a4 ne peut intégrer le groupe car ses produits se situent en dehors de la boule du groupe*

Départ volontaire d'un agent simple d'un groupe Un agent simple peut quitter le groupe ou être « éjecté » du groupe. L'agent simple appartenant à un groupe examine sa situation dans le groupe. S'il estime que la production du groupe est trop éloignée pour son rôle, il peut tenter de quitter le groupe. Il doit faire la demande au groupe. Celui-ci peut accepter ou refuser (voir figure 5.3.9). La raison initiale de ce refus possible était de régler des problèmes de synchronisation lors de la fusion de groupes (un agent simple ne peut quitter son groupe si ce dernier a entamé une procédure de fusion). Cependant le fait que le groupe ait le dernier mot pour le départ d'un agent rend possible la « fixation » des groupes. Cela est une caractéristique intéressante qui permet de figer quelque peu la structure d'une organisation à un certain moment. Ceci peut-être réalisé par l'envoi des messages spécifiques au groupe que l'on désire figer. Lorsqu'un groupe est figé, il ne peut plus éjecter d'agents. Si son niveau d'énergie devient nul, il se met globalement en veille, ainsi que tous ses agents le constituant (de toute façon, ces derniers ne peuvent plus interagir directement avec les autres agents en dehors du groupe).

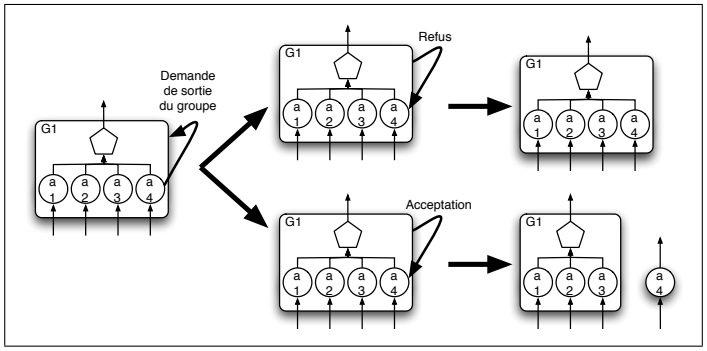


Figure 5.3.9 – *Sortie volontaire d'un agent simple d'un groupe. Le groupe a le dernier mot.*

Éjection d'un agent d'un groupe Inversement, un groupe peut éjecter un agent contenu si il estime que l'agent est en dehors de sa boule dans l'espace d'objets, ou parce qu'il estime qu'il ne contribue pas suffisamment au groupe en terme d'énergie (figure 5.3.10). Lors d'une éjection, l'agent groupe envoie un message d'éjection à l'agent concerné. L'agent aspectuel en question se retrouve sans groupe. Il peut toutefois refaire une demande d'intégration au groupe ou bien à un autre groupe.

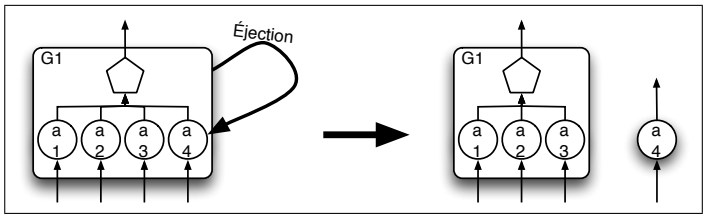


Figure 5.3.10 – *Éjection d'un agent d'un groupe*

Fusion de groupes d'agents En plus de son rôle agrégateur, un groupe peut fusionner avec un autre groupe. L'intérêt d'une fusion est de former un groupe plus gros et donc certainement plus puissant. La fusion de groupes se passe de manière semblable à l'intégration d'un agent simple dans un groupe (voir figure 5.3.11). Un agent groupe qui souhaite fusionner avec un autre groupe initie la procédure en envoyant une demande de fusion. Cette demande contient la liste des produits du groupe. Le groupe obtenu se modifie donc, ainsi que son rôle implicite. Les produits obtenus par le groupe ne sont plus forcément les mêmes. La boule du

groupe dans l'espace des objets se translate (voir figure 5.3.7). Il se peut alors que certains de ses agents se retrouve à l'extérieur de cette boule, et se fassent ainsi éjecter. Il se peut aussi que des agents aspectuels estiment que le groupe n'est plus dans leur boule et décident alors de procéder à un départ volontaire.

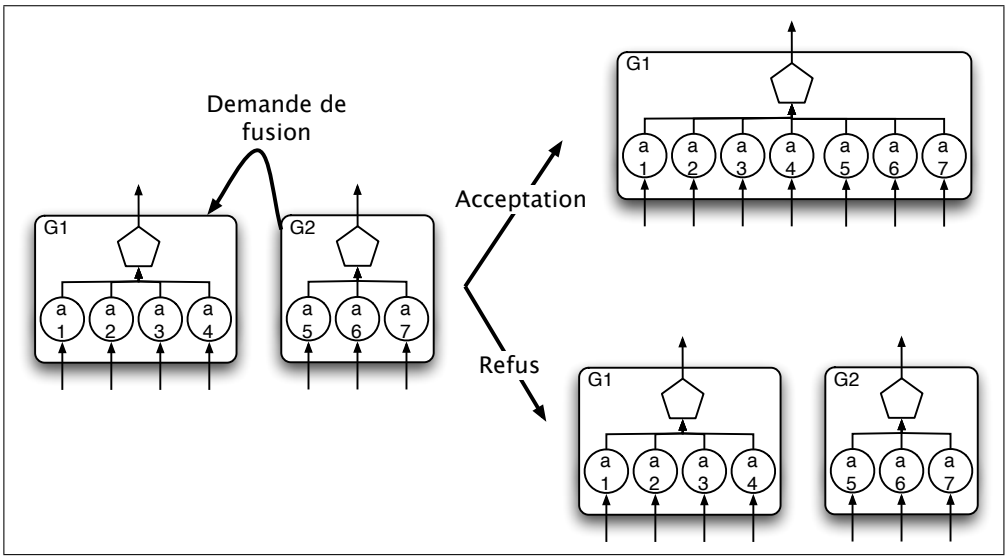


Figure 5.3.11 – *Fusion de groupes. La fusion peut ou peut ne pas se produire.*

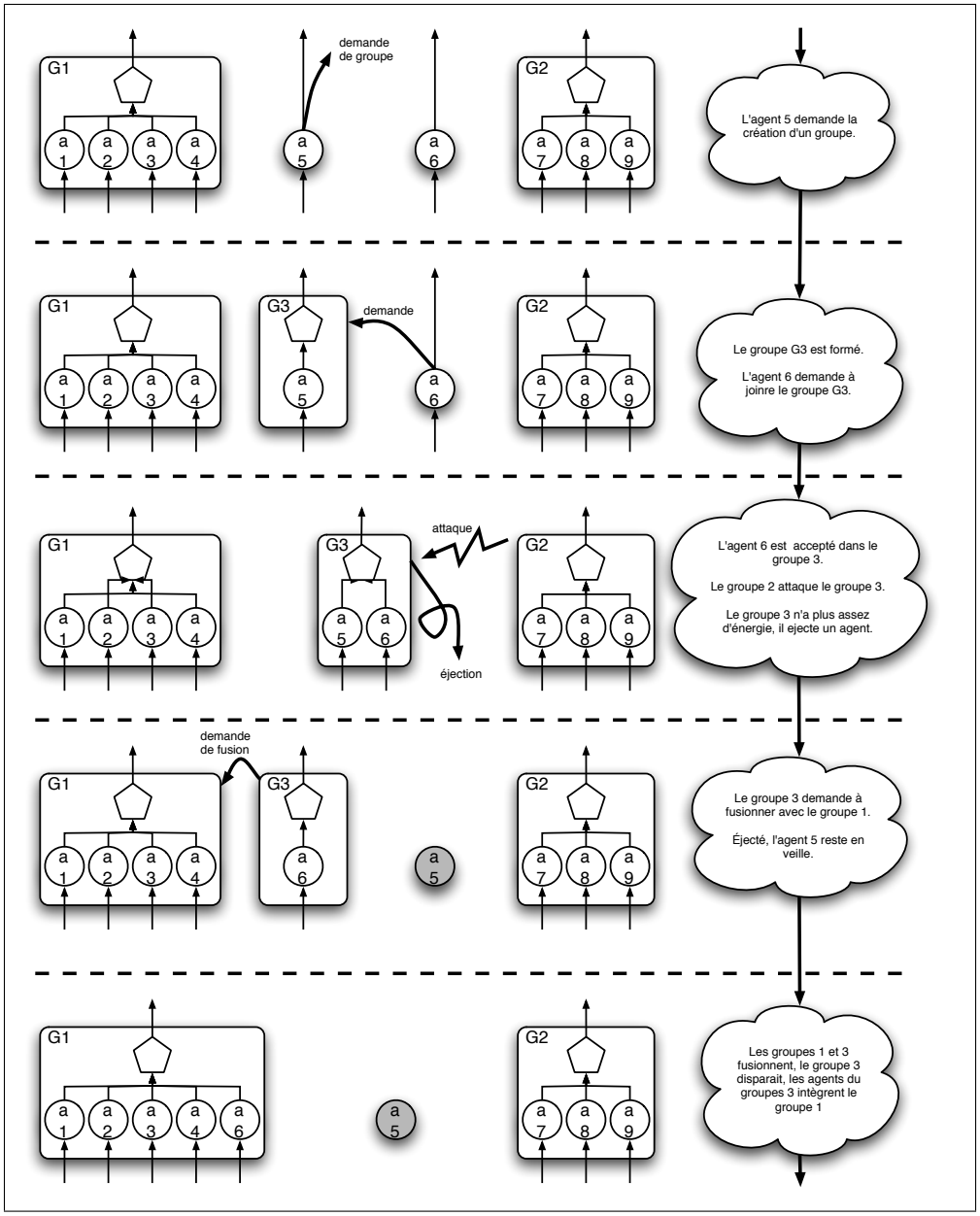


Figure 5.3.12 – *Ce schéma illustre une séquence de quelques événements typiques qui peuvent avoir lieu dans l'organisation.*

5.4 Outil de visualisation

Afin de mettre au point le système, nous avons développé une outil de visualisation. La figure 5.4.1 reproduit une capture d'écran de cet outil à un moment de la simulation, la figure 5.4.2 est un agrandissement de la vue avec légende.

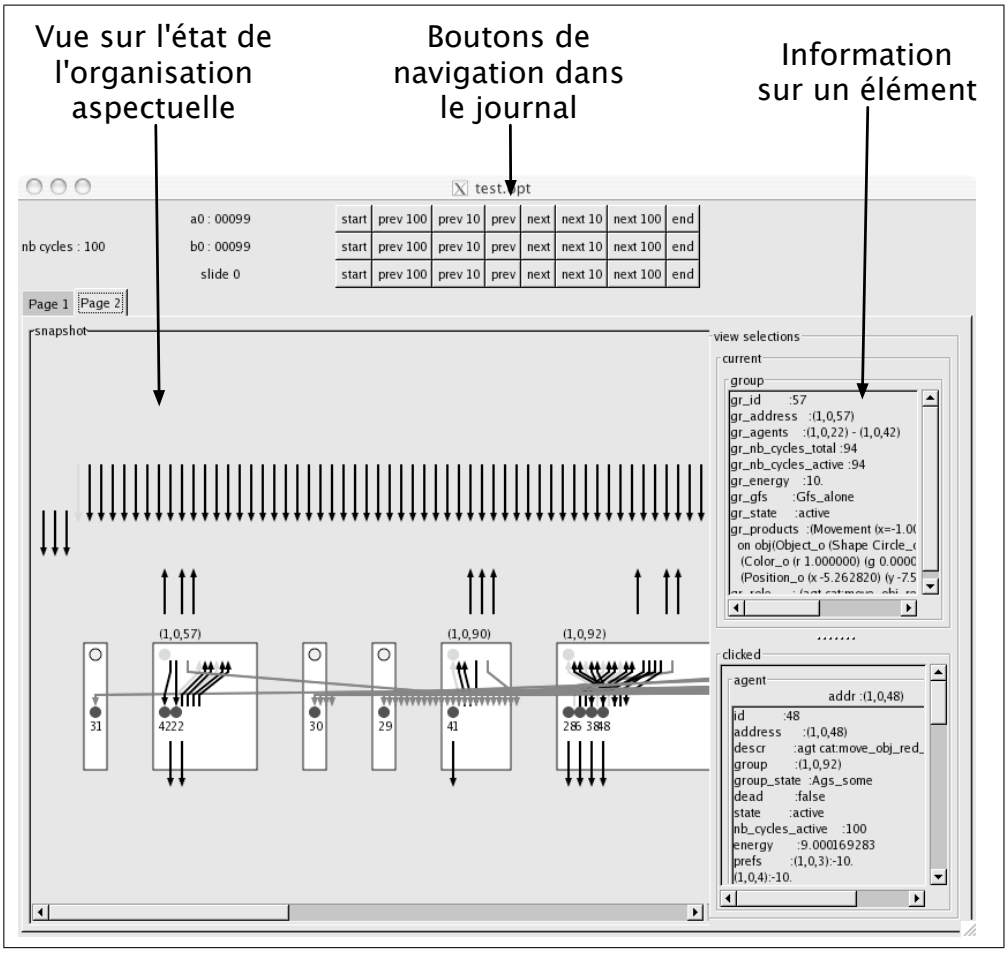


Figure 5.4.1 – *Exemple de visualisation à l'aide du dévermineur graphique. Les caractéristiques des agents et des messages peuvent être affichées sur la droite en passant le curseur de la souris sur chacun des éléments graphiques.*

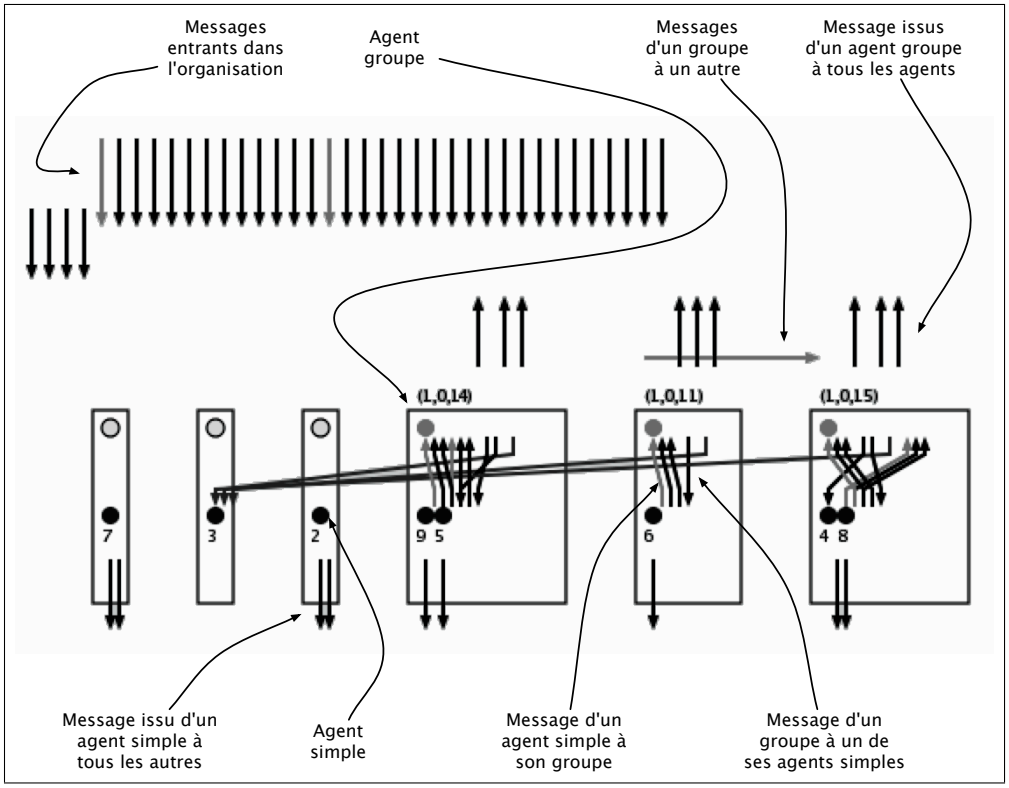


Figure 5.4.2 – *Les agents sont représentés par de petits disques. Les groupes sont matérialisés par un rectangle avec un disque plein en haut. Les messages sont représentés par des flèches. Les flèches allant d'un disque à l'autre sont des messages entre les agents. Les flèches du haut et du bas sont des messages destinés à tous les agents et des messages provenant des agents à destination de tous les autres respectivement.*

Une telle représentation est utile pour mettre au point le système. Cependant le nombre d'agent ne peut pas être très élevé. L'outil est exploitable pour des organisations comportant quelques dizaines d'agents. Lorsqu'il y a davantage d'agents, une telle présentation n'est plus utile, l'utilisateur croule sous le nombre de messages (voir en annexe). Le besoin d'autres outils d'analyse se fait ressentir très vite.

5.5 Remarques

Le système que nous avons présenté est un système complexe. Les entités élémentaires, les agents aspectuels qui le composent, interagissent de façon « relativement autonome » pour produire le comportement global du système. Ce comportement, sauf dans des cas particuliers, est difficile à prévoir *à priori*. Les agents ont un comportement local, ils s'intéressent à l'expression de leur rôle et ne sont pas nécessairement coopératifs. Au contraire la régulation du système est fondée sur l'existence explicite de conflits entre agents ayant des buts opposés. Les phénomènes de formation de groupes, de réduction de certains agents ou le renforcement d'autres font que le comportement global du système exhibe des fonctions différentes. On peut voir ce système comme une machine dont la structure se modifie et par conséquent, transforme sa fonction. C'est un système qui a la possibilité de se reconfigurer et de produire des fonctionnalités différentes et émergentes. Bien que toutes les possibilités sont implicites, elles émergent par la structuration de l'organisation au cours du temps.

Cependant, on doute qu'un tel système, où il existe des agents de rôles différents et opposés n'ayant pas forcément les même buts, et livré à lui même, puisse opérer de façon adéquate. Peut-être que dans certains cas le système tendra spontanément vers une structure satisfaisante, mais en général, de part la nature des agents le composant, le système sera relativement instable et il sera nécessaire de forcer, de temps en temps, sa structure ou certaines parties clefs, pour le faire « converger » vers une forme adéquate. Il faut donc absolument introduire un mécanisme de pilotage, qui entretienne une tendance émergente, au moins un certain temps. Un tel système est ainsi un substrat intéressant pour l'application de notre modèle de contrôle morphologique.

5.6 Morphologie et analyse du système

Dans cette section nous allons présenter la méthode retenue pour représenter et analyser le système introduit précédemment. La méthode choisie est fondée sur la représentation de l'organisation d'agents par un graphe. Cette représentation est relativement naturelle.

5.6.1 Proposition de morphologie : graphes

Le système décrit ci-dessus est plus complexe que le premier exemple du chapitre 4. Il permet de mettre en œuvre des outils d'analyse plus sophistiqués. Néanmoins, une analyse analogue à celle utilisée dans le premier exemple, une étude de la distribution d'une caractéristique au sein de l'organisation, serait envisageable. Nous allons exploiter la sémantique que l'on retrouve au niveau des rôles des agents. L'organisation aspectuelle peut être considérée comme un réseau dynamique d'agents. Nous proposons d'analyser la structure et l'évolution de ce réseau par analyse de la forme du graphe le représentant. L'outil d'analyse morphologique sera une projection de l'organisation dans l'ensemble des graphes orientés définis de la façon suivante : les nœuds correspondront aux agents plongés dans l'espace des rôles et les arcs correspondront aux préférences mutuelles des agents.

5.6.2 Analyse de graphes

Méthodes existantes d'analyse de graphes

Comme nous avons vu dans l'exposé général du modèle, au chapitre 3, l'étude morphologique requiert un moyen pour comparer des formes. Ici, il faudra comparer des graphes.

D'une manière générale, les graphes sont utilisés pour représenter mathématiquement de la connaissance structurelle. Ce type de représentation est de plus en plus utilisé, dans de nombreux domaines, ceci grâce en partie à la possibilité d'utiliser ces graphes en combinaison avec des algorithmes d'appariement. Cette représentation permet de représenter les différences et les variations structurelles entre des objets différents. Différentes applications ont été proposées en utilisant ce type de représentation pour réaliser par exemple, la reconnaissance automatique des images [Ben02]. Dans ce type d'application, la connaissance sur les variations structurelles possibles des objets qui apparaissent dans les images est exprimée à travers un modèle que l'on représente comme un graphe (par exemple dans la cartographie, la reconnaissance de caractères, et la reconnaissance de structures cérébrales à partir d'images de résonance magnétique...).

En notant $G_M = (V_M, E_M)$ le graphe modèle et $G_D = (V_D, E_D)$ le graphe de données, le problème consiste à trouver un homomorphisme $h : V_D \mapsto V_M$tel que $(u,v) \in E_D$ si et seulement si $(f(u), f(v)) \in E_M$. Ces homomorphismes sont évalués en calculant la similarité entre les attributs des noeuds et des arêtes mis en correspondance. On cherche alors un meilleur homomorphisme au sens d'un certain critère combinant ces similarités.

Beaucoup de travaux se consacrent à la recherche d'isomorphismes de graphes, ce qui suppose en particulier que $|V_M| = |V_D|$. On parle alors de correspondance exacte. Cependant, dans beaucoup de problèmes, surtout pour ceux où la segmentation s'effectue automatiquement, la condition d'isomorphisme est trop stricte et le nombre de noeuds du graphe de données est supérieur à celui du modèle ($|V_M| < |V_D|$). On parle alors de correspondance inexacte de graphes, et la complexité de ces problèmes est supérieure à celle des précédents. De nombreuses techniques très différentes ont été développées pour la mise en correspondance de graphes : arbres de décision [MB00], réseaux de neurones [DJD+un], algorithme EM [CH98], relaxation probabiliste [CKP95], heuristiques et méta-heuristiques [PSZ99], algorithmes d'estimations de distributions [Ben02], algorithmes génétiques [WH97], et programmation évolutive [SCC97].

Remarques

Les techniques existantes sont assez performantes en termes de résultats, mais sont très gourmandes en ressources de calcul et également en temps de calcul. Le problème de comparaison des graphes est un problème complexe de type NP dans le cas général. Soucieux de faire tourner notre système en un temps relativement raisonnable, nous avons opté pour une approche approximative.

5.6.3 Types de graphes, hypothèses

Comme nous l'avons vu plus haut, l'organisation d'agents est un ensemble structuré d'agents possédant chacun un rôle et des caractéristiques propres, et ces agents sont en relation. Les graphes représentent une facette de la structure de l'organisation d'agents. Nous avons choisi de représenter la façon dont les rôles des agents sont en relation.

Les nœuds des graphes représentent les agents, agents simples et agents groupes. Les attributs des nœuds représentent en partie le rôle des agents. Pour les agents simples il s'agit de leur rôle propre. Pour les agents groupes, il s'agit de leur rôle issu de la pondération des rôles des agents simples les constituant. Les arcs entre les sommets représentent la relation de préférence entre les agents. Cette relation est dirigée et n'est pas forcément symétrique.

5.6.4 Construction des graphes

À chaque cycle de la simulation, les informations sur les agents sont récupérées par le module morphologique (voir figure 5.6.1). Celui-ci extrait celles qui sont nécessaires à la construction du graphe, à savoir le rôle de chaque agent. Pour les agents simples, le rôle est défini explicitement (voir section 5.2.5). Pour les agents aspectuels groupes, le rôle de l'agent est défini par l'ensemble des rôles des agents le composant (voir section 5.2.5).

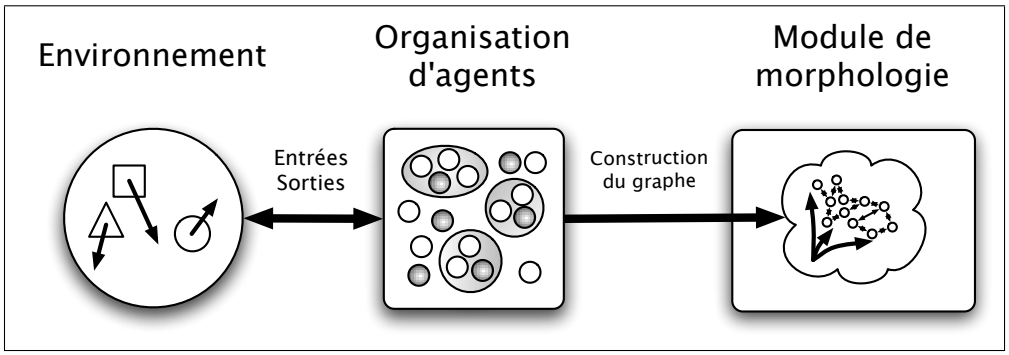


Figure 5.6.1 – *Construction du graphe : le graphe représentant la structure de l'organisation aspectuelle est construite par le module de morphologie à partir des mesures effectuées sur les agents aspectuels. Le graphe est construit à chaque cycle de la simulation.*

Considérons l'ensemble des rôles des agents $S = (s_1, \ldots, s_m)$ (« S » pour « sémantique ») à m dimensions. À chacun de ces rôles, ou dimensions, nous associons un axe (intervalle $I = [0;1]$). Nous obtenons alors un espace I^m. Dans cet espace nous pouvons positionner chacun des nœuds (ou agent) du graphe en fonction de sa proportion de rôles relative (voir figure 5.6.2). De plus, afin de rendre les

calculs encore plus rapides, (moyennant encore une approximation), nous discrétisons l'espace I^m, chacun des axes est discrétisé en p classes de tailles identiques.

On peut voir cette représentation un peu comme un histogramme multi-dimensionnel auquel on ajoute des relations entres les différentes classes.

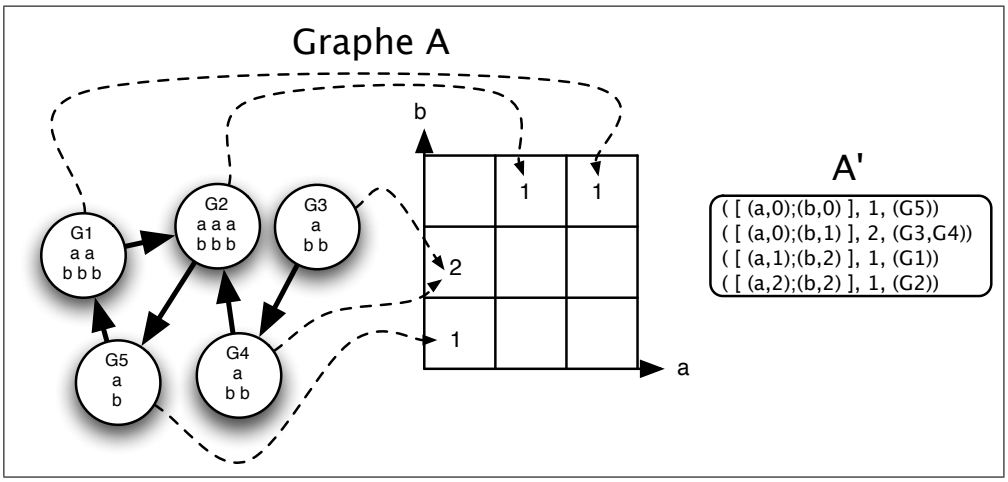


Figure 5.6.2 – *Construction d'une représentation approchée du graphe. Le graphe est plongé dans un espace indexé. Cette approximation permet de faire des comparaisons, approchées mais rapides, de graphes.*

5.6.5 Comparaison entre graphes

Le principe de calcul de différence entre les graphes repose sur à peu près la même idée que l'on retrouve dans une grande partie des méthodes de calcul de distance de graphes. On détermine d'abord les modifications à apporter à un graphe pour le transformer en un autre et la distance entre les graphes est déterminée en attribuant des coûts à chacune des modifications. Dans les techniques exactes, la distance est définie comme étant le coût minimal. Pour notre part, nous proposerons une approximation de cette méthode, toujours dans le souci d'effectuer les calculs en temps raisonnable. C'est pour cela que nous utiliserons le terme de « pseudo-distance ». Nous aurions pu mettre en œuvre des techniques plus « justes », ce qui est envisagé dans des recherches futures. La différence supplémentaire est que nous ne travaillons pas directement sur les graphes, mais sur leur approximation discrétisée.

Une fois les approximations discrétisées des graphes calculées (G'_1 et $G'2$), nous effectuons la différence paire à paire des classes pour calculer un troisième histogramme multidimensionnel, noté $G' = G'_1 - G'_2$ (voir figure 5.6.3). À partir de ce troisième histogramme, nous énumérons trois types de classes : celles à coefficient nul, celles à coefficient positif et celles à coefficient négatif (en ignorant celles qui ne font pas intervenir d'agents). Les classes à coefficient nul, notée C_0, correspondent aux classes où il y a autant d'agents de même rôle dans chacun des deux graphes comparés. Celles à coefficient positif (C_+) et négatif (C_-) correspondent respectivement aux classes où les agents sont en excès et en défaut d'un graphe à l'autre.

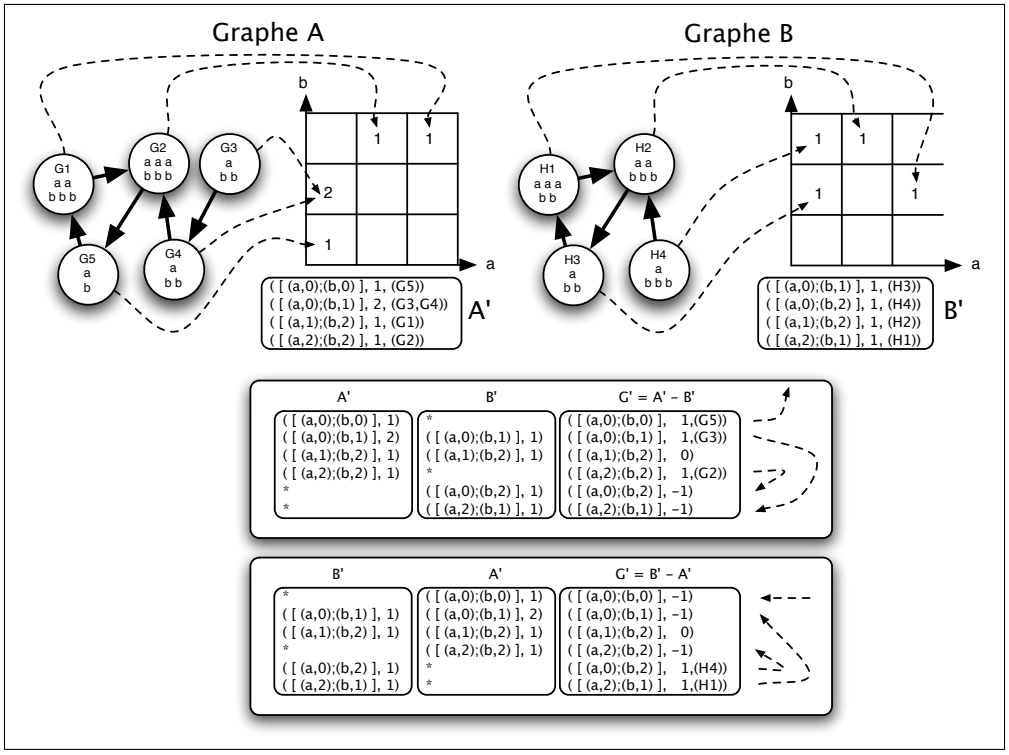


Figure 5.6.3 – *La comparaison approchée (G') de graphes s'effectue à partir des représentations approchées (A' et B') des deux graphes (A et B) à comparer.*

À partir de cela, on calcule les modifications à apporter à un graphe pour le transformer en le second graphe. Un ensemble de « migrations » d'agents est spécifié à partir des classes C_- et C_+. Ce sont en fait des ordres qui seront envoyés aux agents intéressés, notamment pour les agents groupes. Pour les agents simples, qui ne peuvent pas changer de rôle, la solution suivante a été adoptée : si

un agent simple reçoit un ordre de changer de rôle, il se met en veille.

Pour les agents de la classe C_0, on calcule les modifications à opérer sur les arcs, paires à paires, pour les faire correspondre (arcs absents, arcs en excès, arcs de pondérations différentes).

Le calcul des modifications étant effectué, nous pouvons associer à chacune des opérations des coefficients pour avoir la pseudo-distance entre les deux graphes. Cependant, nous choisissons de ne pas prendre en compte les différences entre les arcs des nœuds qui ne se trouvent pas dans la classe C_0 dans un premier temps.

5.6.6 Module d'analyse

Le module morphologique fournit donc une représentation géométrique, synthétisée sous une forme manipulable. Pour boucler le système, afin de pouvoir le contrôler s'il le faut, le module d'analyse va construire le modèle du système tel que cela a été décrit dans la section 3.5 (voir figure 5.6.4).

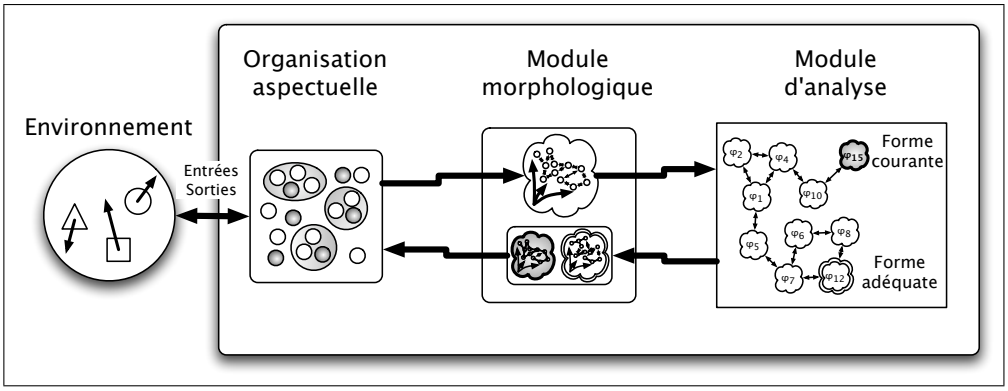


Figure 5.6.4 – *La boucle de contrôle de l'organisation aspectuelle fondée sur l'analyse morphologique à base de graphe.*

5.6.7 Retour sur le système

Nous avons vu que la structure de l'organisation a été représentée par un graphe. Supposons que le module d'analyse souhaite faire tendre l'organisation aspectuelle, qui se trouve sous une forme particulière (graphe G_1) vers une autre forme. Le module d'analyse propose une forme (un graphe G_2) au module morphologique qui calculera les différences et les transformations à effectuer. Le module morphologique enverra des ordres aux agents aspectuels concernés. Notons que les agents aspectuels simples ne peuvent changer de rôle, mais ils peuvent se regrouper pour former des groupes de rôles un peu différents. Quand aux agents groupes, ils peuvent soit intégrer des agents simples supplémentaires ou en éjecter certains pour faire migrer leur rôle implicite. Les agents groupes peuvent également plafonner la contribution d'un agent aspectuel simple, de manière à faire changer son rôle. L'agent simple, ainsi limité, pourrait éventuellement quitter le groupe si le groupe diverge trop à son goût.

Chapitre 6

Application

Dans cette partie nous décrivons un cas d'application du système décrit dans le chapitre précédent. Nous commencerons pas exposer l'environnement, le système, l'ontologie et les rôles des agents (section 6.1). Nous exposerons quelques résultats préliminaires (section 6.3).

6.1 Description de l'application

Le système est interfacé à un environnement planaire simulé où se situent des objets de formes élémentaires et de couleurs différentes (voir figure 6.1.1).

Le système a la possibilité de déplacer les objets en envoyant des ordres à l'environnement. Les objectifs du système sont spécifiés sous formes de règles. Ces règles sont utilisées pour l'apprentissage du système, comme par exemple bouger les objets dans une certaine direction.

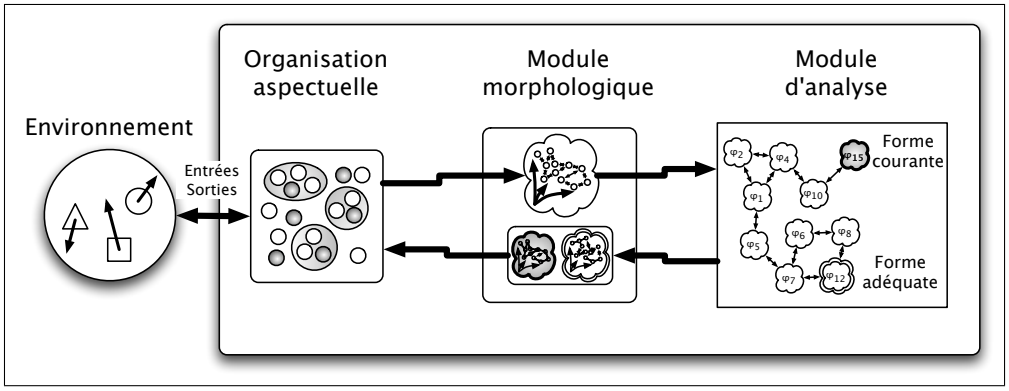


Figure 6.1.1 – *Schéma de l'application. Le système peut agir sur l'environnement où se trouvent des objets qu'il peut bouger. Le système est piloté par une organisation aspectuelle.*

6.1.1 Les objets de l'environnement

Les objets de l'environnement sont décrits de façon symbolique. Un objet o est défini par :

- sa forme f ;
- sa couleur c ;
- sa position p.

Une forme f est soit :

- un carré, défini par la longueur c (dans $\mathbb{R}$) du côté ;
- un rectangle, défini par la longueur l_1 (dans $\mathbb{R}$) et l_2 (dans $\mathbb{R}$) des deux cotés ;
- un cercle défini par le rayon r (dans $\mathbb{R}$) ;
- une ellipse définie par les deux rayons r_1 (dans $\mathbb{R}$) et r_2 (dans $\mathbb{R}$).

Une position p dans le plan est définie par un couple :

- l'abscisse x dans $\mathbb{R}$;
- l'ordonnée y dans $\mathbb{R}$.

Une couleur c est définie par un triplet de façon classique :

– composante rouge r dans $[0;1]$;
– composante verte v dans $[0;1]$;
– composante rouge b dans $[0;1]$.

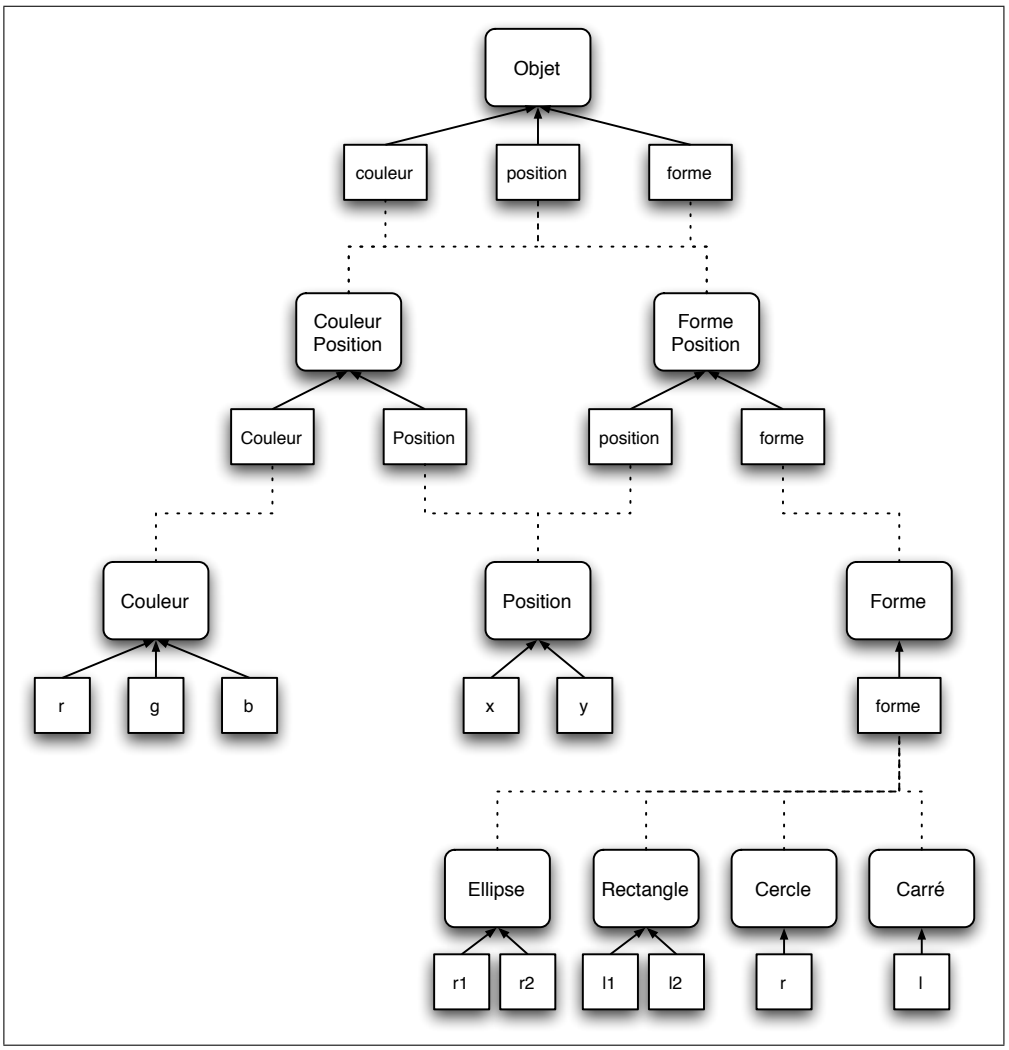


Figure 6.1.2 – *Ontologie*

6.2 Agentification

6.2.1 Rôle des agents

Nous avons inclus uniquement des agents de mouvement dans une première version de l'exemple et nous nous sommes intéressés au comportement du système dans ce cas.

Le rôle des agents sera du type (voir figure 6.2.1) :

- pour l'activation : détection d'un objet répondant à un critère dans une zone de l'environnement donné ;
- pour la production de symboles : suggérer un déplacement de l'objet en question dans une direction donnée.

Plus spécifiquement, les rôles des agents dans cet exemple sont :
- si un objet se trouve en haut ($y > -5$), le déplacer vers le bas ;
- si un objet se trouve en bas ($y < 5$), le déplacer vers le haut ;
- si un objet se trouve à droite ($x > -5$), le déplacer vers la gauche ;
- si un objet se trouve à gauche ($x < 5$), le déplacer vers la droite.

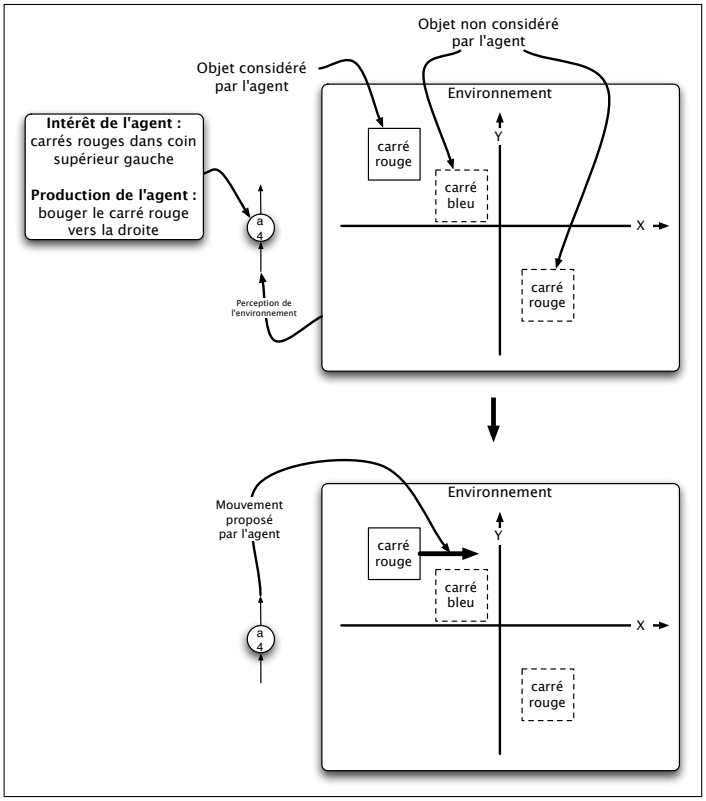


Figure 6.2.1 – *Propositions de mouvement individuel de l'agent dont le rôle est de pousser vers la droite les objets rouges qu'il détecte dans la zone en haut à gauche de l'environnement.*

Nous avons choisi de faire se recouvrir les zones d'activation des rôles des agents. Cela se traduit pas les conditions : $y > -5$, $y < 5$, $x > -5$ et $x < 5$ à la place des conditions : $y > 0$, $y < 0$, $x > 0$ et $x < 0$. Nous avons choisi ces règles pour que des agents de rôles opposés puissent s'activer en même temps et entrer en conflit. Par ailleurs, cela laissera les objets au voisinage de l'origine du repère, les objets resteront dans le carré $((-5,-5);(5,5))$

Un mouvement est défini comme un déplacement se référant à un objet en particulier :

```
type movement = {
  movement__object : object_o ;
  movement__x      : float ;
```

```
  movement__y       : float ;
} ;;
```

6.2.2 Connaissances réifiées dans les agents

Les connaissances de comparaison (au niveau des agents aspectuels simples) et d'agrégation (au niveau des agents groupes) des mouvements d'objets sont réifiées dans les agents par le biais des fonctions de comparaison et d'agrégation définies ci-dessous.

Règles de composition

De façon analogue, la fonction d'agrégation d'un mouvement est définie de la manière suivante :

Pour deux mouvement $m_1(o, dx_1, dy_1)$ et $m_2(o, dx_2, dy_2)$, on définit la fonction d'agrégation suivante :

$m(o, dx, dy) = f_{agr}(m_1, m_2)$ avec $dx = \frac{(c_1 dx_1)+(c_2 dx_2)}{c_1+c_2}$, $dy = \frac{(c_1 dy_1)+(c_2 dy_2)}{c_1+c_2}$

Métrique

La comparaison des mouvements se fait à l'aide de la métrique suivante : pour $m_1(o, dx_1, dy_1)$ et $m_2(o, dx_2, dy_2)$: $d(m_1, m2) = \sqrt{(dx_1 - dx_2)^2 + (dy_1 - dy_2)^2}$

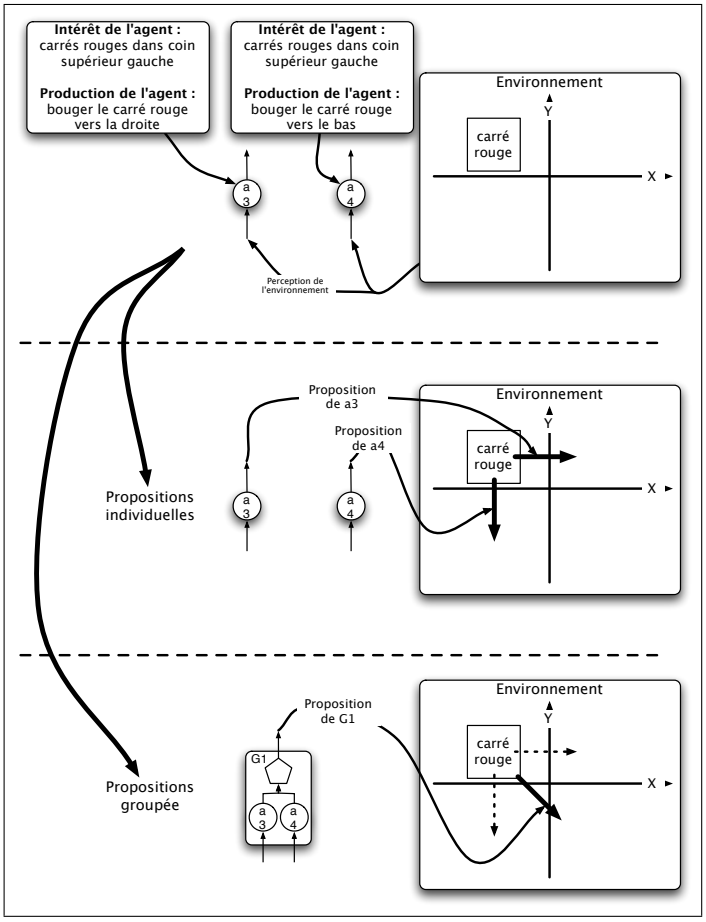


Figure 6.2.2 – *Propositions de mouvement avec groupe*

Selon la structure de l'organisation, le comportement global du système est différent (voir figure 6.2.3).

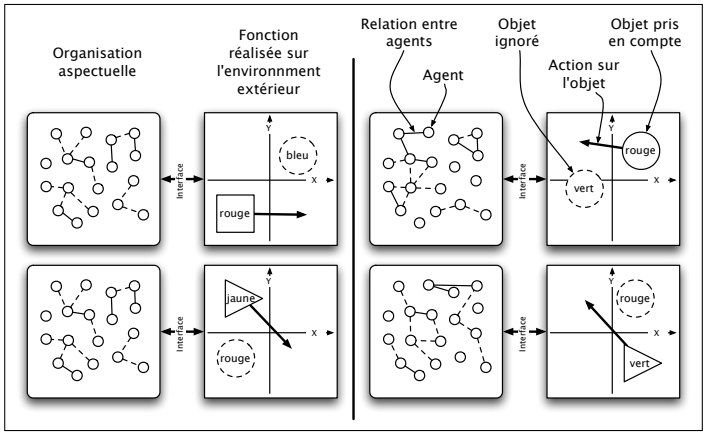


Figure 6.2.3 – *La structure de l'organisation détermine la fonction du système sur l'environnement.*

Le rayon de préférence des agents a été fixé à $1,5$. Ainsi les agents proposant des déplacements, considérés comme des objets dans le monde des agents et dont la distance est inférieure à $1,5$, pourront s'agréger pour former un groupe. Nous avons choisi $1,5$, car ce seuil est supérieur à $\sqrt{2}$, laissant ainsi aux agents proposant des mouvements orthogonaux, la possibilité de s'allier, mais pas aux agents proposant des mouvement opposés.

6.3 Quelques résultats

Nous présentons ici quelques résultats préliminaires sur l'étude de ce système. Cependant davantage d'expérimentations seront nécessaires pour mieux comprendre son fonctionnement. Les résultats indiqués sont une moyenne sur un ensemble d'expériences avec des valeurs de paramètres différents. Certaines de ces expériences ont été menées avec 40, 80 et 120 agents simples. La principale limitation au nombre d'agents est due à l'architecture du prototype. En effet, le prototype a été conçu pour mettre au point le modèle, et dans cette optique, les diverses structures de données nécessaires étaient assez gourmandes en mémoire.

La figure 6.3.1 illustre un exemple simple d'environnement où se trouve un seul objet, un cercle rouge.

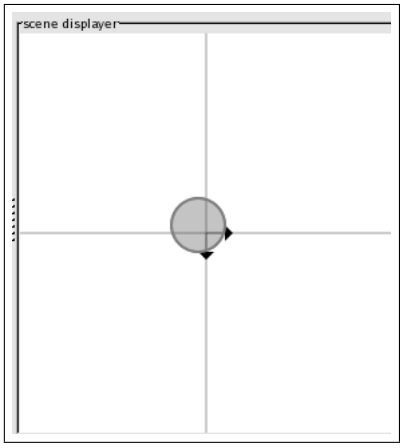


Figure 6.3.1 – *Visualisation de l'objet dans l'environnement. Il s'agit dans cet exemple d'un cercle rouge. Les lignes horizontales et verticales représentent les axes du système de coordonnées de l'environnement. Les flèches alignées sur les axes représentent les bases du repère. Le centre du cercle est placé dans le troisième quart du plan.*

La figure 6.3.2 illustre quelques exemples de mouvements proposés par le système. On peut y observer des mouvements de même direction que les axes, qui correspondent aux mouvements de base que l'on retrouve au niveaux des agents simples et éventuellement dans les agents groupes constitués d'agents de même rôles. Par ailleurs, on observe également des mouvements non parallèles aux axes du plan. Ces mouvements sont issus d'agents groupes formés d'agents simples ayant des rôles différents, comme cela est illustré sur la figure 6.2.2. Les mouvements issus de l'agrégation d'agents ne sont pas forcément alignés selon les bissectrices du plan. En effet, il se peut que la contribution des agents au sein d'un groupe ne soit pas homogène (l'agent simple souhaitant déplacer l'objet à droite donne plus d'énergie au groupe que l'agent simple souhaitant déplacer l'objet vers le bas). Il se peut également que le nombre des agents simples d'un rôle soit différent de celui des agents d'un autre rôle.

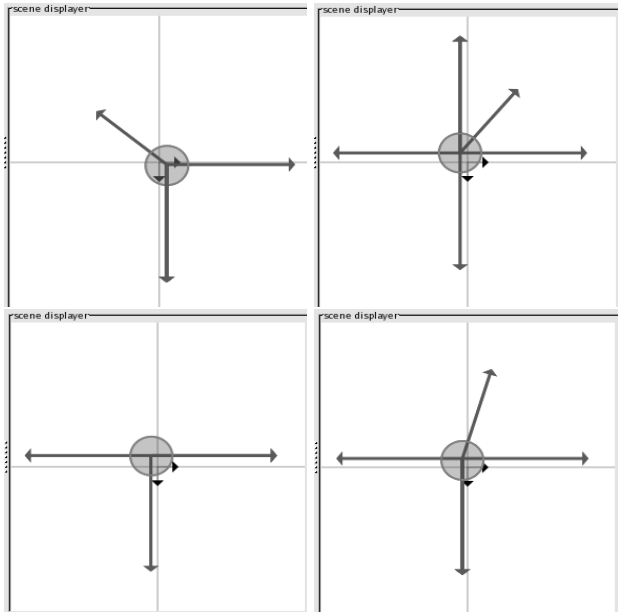


Figure 6.3.2 – *Exemple de mouvements suggérés par le système pour l'objet situé dans l'environnement. Les mouvements non parallèles aux axes correspondent aux mouvements proposés par des agents groupes, issus le l'agrégation d'agent de rôle simple.*

En fait, de par la symétrie du système et du nombre redondant d'agents de même rôle, il se peut que plusieurs agents différents proposent le même mouvement. Cela ne se voit pas sur la figure 6.3.2 car tous les mouvement identiques se superposent. Néanmoins, il est possible à l'affichage, de déplacer un peu chacun de ces mouvements proposés afin de les distinguer, comme sur la figure 6.3.3.

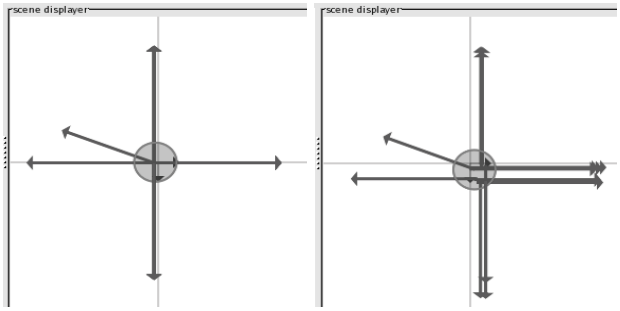


Figure 6.3.3 – *Plusieurs agents (simples ou groupes) peuvent proposer le même type de mouvement. Sur la figure de gauche, on ne peut voir cela car les mêmes mouvements sont superposés. En revanche, en « éparpillant » l'affichage des mouvements (en ajoutant du bruit dans leur positionnement à l'écran) on peut les distinguer.*

La figure 6.3.4 illustre le cas avec deux objets : un cercle rouge et un carré bleu placés à des endroits différents de l'environnement. On peut noter que les mouvements appliqués par le système sur ces objets ne sont pas les mêmes. En effet, les agent simples, dans ce cas, s'intéressent à un objet en particulier. Les agents perçoivent tous les objets de l'environnement. Ils proposent pour chacun des objets un mouvement en fonction de leur rôle. Cependant les agents ont été paramétrés pour n'émettre qu'un seul mouvement. Ils choisissent au hasard le premier mouvement qu'ils émettent. Par la suite ils calculeront toujours des mouvements pour tous les objets, mais n'émettront que celui qui sera le plus proche de celui qui a été émis précédemment. Ce choix a été fait pour donner au système une certaine inertie et de la continuité.

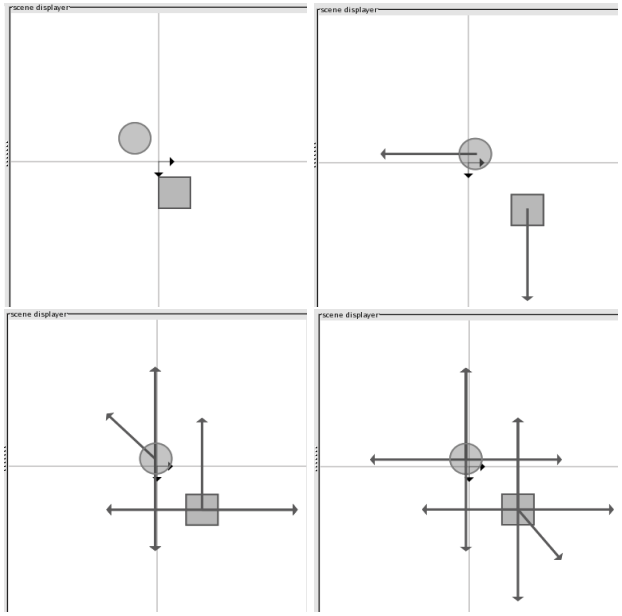


Figure 6.3.4 – *Exemple avec deux objets dans l'environnement. Les objets n'ont pas les mêmes déplacements. Les agents ne sont pas tous intéressés par les mêmes objets.*

6.3.1 Expériences

Nous avons d'abord voulu tester le système dans des configurations simples.

Sans contrôle

La figure 6.3.5 illustre un exemple typique de déplacement d'un objet au cours du temps (1000 itérations sur cette figure). L'objet est placé initialement en $(0,0)$. Au cours du temps, il évolue dans le plan, soumis aux mouvements proposé par le système. Nous avons remarqué qu'un fois que l'objet est poussé vers un coin ou un côté, il a de fortes chances d'y rester. Il semblerait que certains agents parviennent à adopter une position dominante dès le départ. Et d'une manière générale, ces agents parviennent à rester sur un côté.

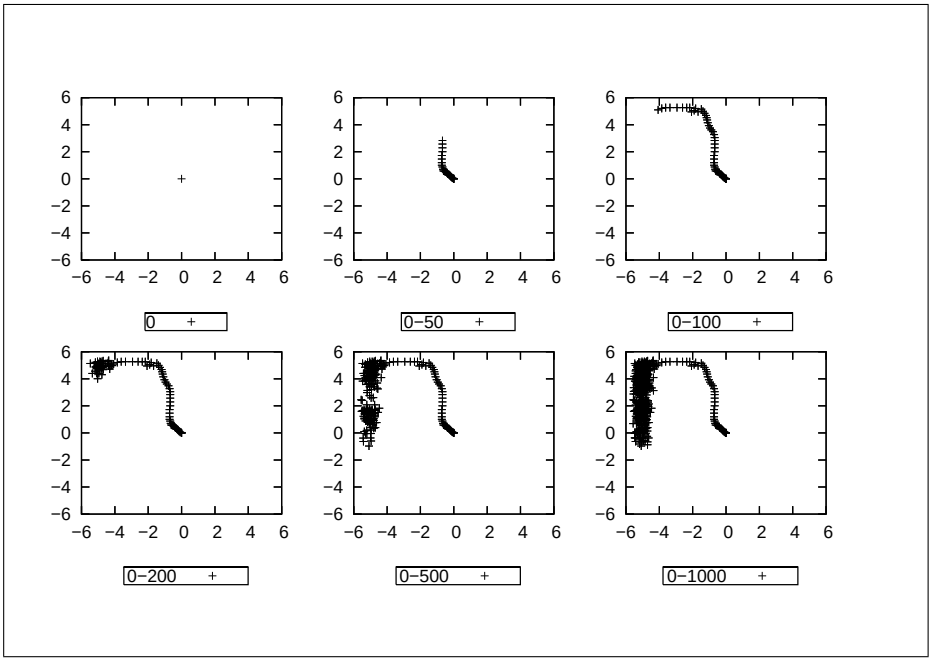


Figure 6.3.5 – *Traces typiques de mouvements de l'objet sans contrôle du module d'analyse. Les points indiquent les positions successives de l'objet au cours de la simulation. Les pas d'itérations pris en compte, sur chaque vignette, sont indiqués en dessous de celles-ci.*

Nous avons effectué une série de tests avec une initialisation différente du générateur de nombre pseudo-aléatoires, pour estimer le comportement moyen du système (toujours sans contrôle du module d'analyse).

Au lancement, le système est dans une configuration particulière. Les attributs des agents, tels leurs préférences et leurs historiques, sont vierges. Le système présente une très haute symétrie, ce qui est dû à l'initialisation. Au fur et à mesure du temps, cette symétrie est brisée par l'interaction entre les agents. Ainsi, pour ne pas « polluer » l'ensemble d'apprentissage du module d'analyse par des états très particuliers, nous avons choisi de faire tourner le système un certain nombre de cycles afin que celui-ci adopte un régime « normal ». C'est après cet état transitoire que nous observons réellement le système (le module d'analyse commence à mémoriser les formes de l'organisation aspectuelle).

La figure 6.3.6 illustre l'extraction d'un graphe d'évolution sur un exemple de simulation comportant un petit nombre d'agents (quelque dizaines). Ce type de

graphe est construit au fur à partir de l'observation de l'organisation aspectuelle par le module d'analyse ainsi que sa description morphologique issue du module de morphologie. Cette séquence illustre les états du graphe aux itérations 101, 151, 175, 211, 271 et 303 sur 1000 itérations. Le module d'analyse n'a plus mémorisé de forme après l'itération 303, et ce jusqu'à l'itération 1000 (la fin de la simulation).

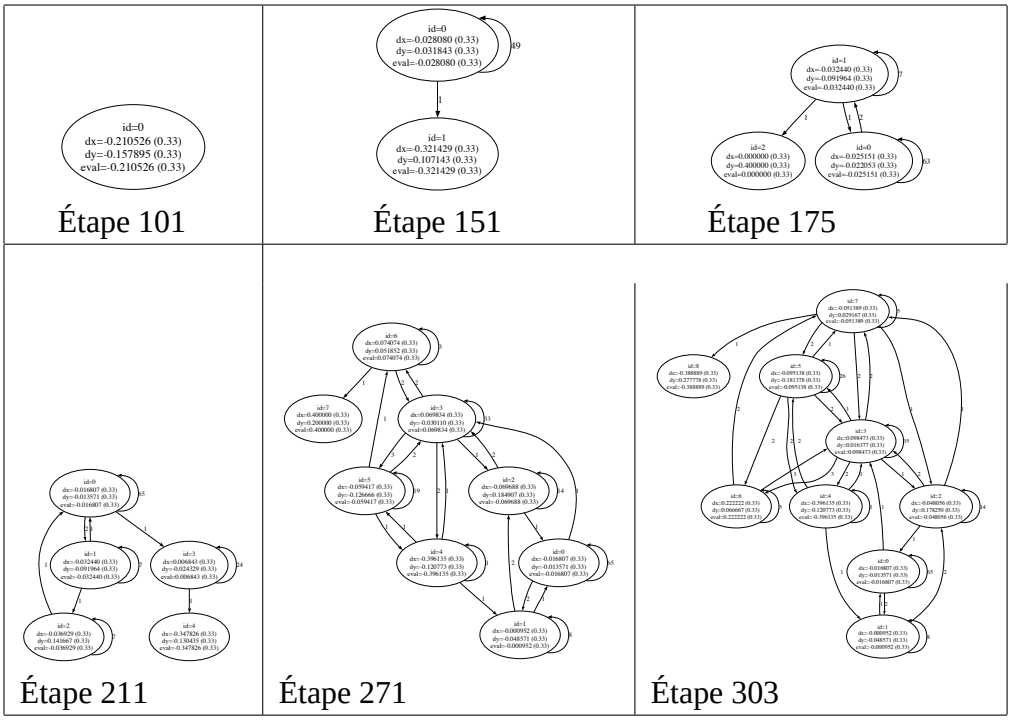


Figure 6.3.6 – *Exemple d'extraction de graphes d'évolution. Ce type de graphe est construit au fur et à mesure à partir des observations du comportement du système et de la description morphologique de la dynamique du système. Chacun des nœuds de ces graphes est une forme. Il s'agit, dans ce cas, de graphes décrivant la répartition des rôles du système dans l'espace des rôles.*

À chaque fois que le système prend une forme qui n'est pas dans l'ensemble d'apprentissage (ou qui n'est pas équivalente à l'une des formes représentées par les nœuds du graphe d'évolution), cette forme est ajoutée à l'ensemble d'apprentissage. Ceci se traduit par une nouvelle transition vers une nouvelle forme sur le graphe d'apprentissage.

Les nœuds et les arcs de ce graphe ont des attributs. Ces attributs évoluent

au cours du temps du fait de l'apprentissage. Les attributs des nœuds, dans cet exemple, sont relatifs au déplacement produit par le système et à l'évaluation de la performance du système. Les attributs des arcs sont le nombre de fois que la transition qu'ils représentent a été empruntée par le système.

La figure 6.3.7 illustre un ensemble de traces de mouvement de l'objet suivant différentes configurations du système (en terme de nombre d'agents, seuil de comparaison des formes, initialisation du générateur de nombres pseudo-aléatoires...). Nous observons que sur un grand nombres d'essais, le comportement moyen du système est de déplacer l'objet à l'écart de l'origine et vers le bord du carré virtuel $(-5,-5)-(5,5)$. Ce bord n'existe bien sûr pas réellement. Les bords correspondent aux frontières des zones d'activation et de désactivation des rôles des agents.

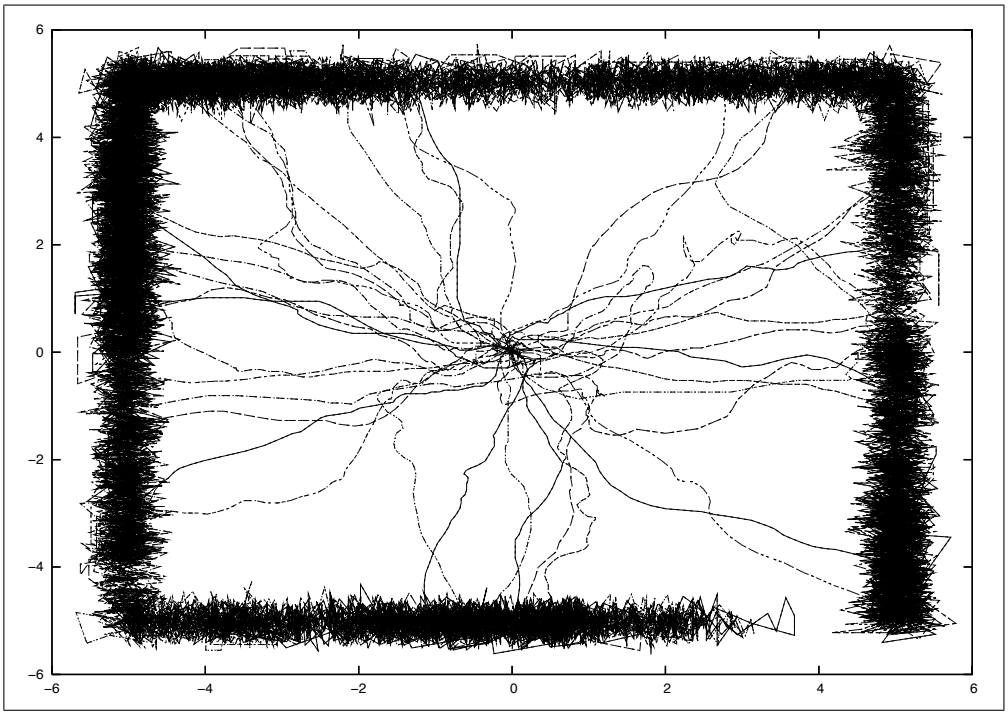


Figure 6.3.7 – *Traces du comportement moyen du système sur un ensemble d'essais. Sans contrôle, le système à tendance à pousser les objets aux bords (virtuels).*

La figure 6.3.8 résume l'évolution au cours du temps de la taille du graphe d'évolution pour différents paramétrages du système. On observe qu'une grande part des transitions ont lieu assez rapidement, entre l'itération 100 et 300. Ce

graphe montre qu'aucune forme n'est apprise avant l'itération 100. En effet, comme nous l'avons précisé précédemment, nous faisons évoluer le système librement pour qu'il quitte l'état particulier dans lequel il se trouve au lancement.

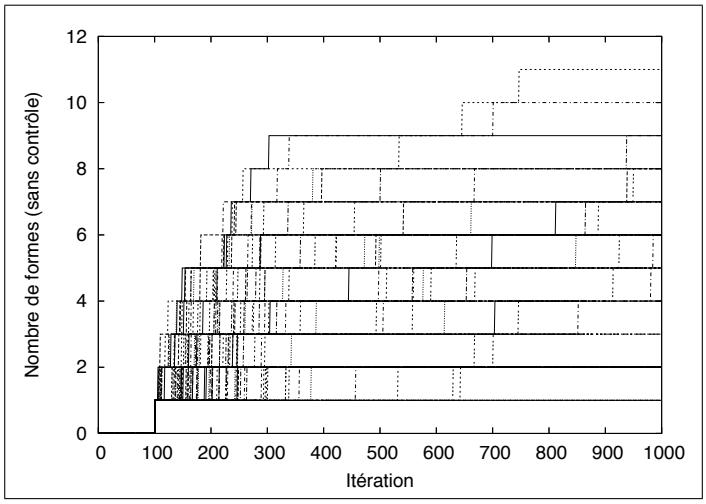


Figure 6.3.8 – *Évolution de la taille du graphe d'évolution au cours du temps pour un ensemble de paramétrages différents du système.*

La figure 6.3.9 illustre la distribution du nombre de formes que prend le système lorsqu'il est libre, pour un ensemble de configurations du système.

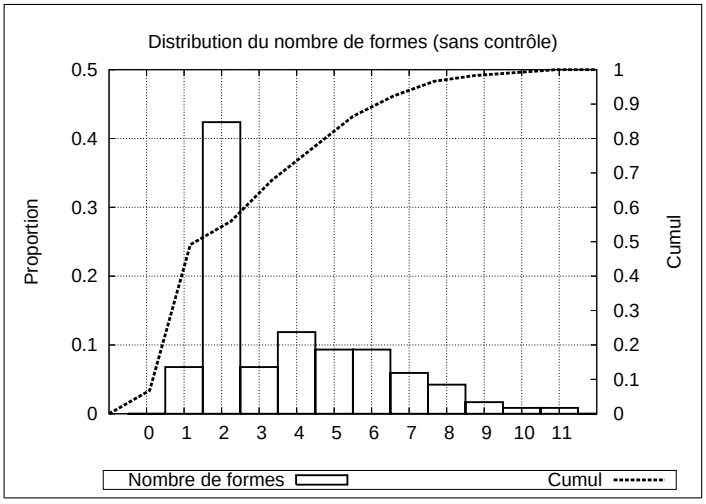


Figure 6.3.9 – *Distributions du nombre de formes apprises pour un ensemble de paramétrages du système (sans contrôle).*

On observe qu'il y a beaucoup de cas ou le système se conforme uniquement sous deux formes. Cela veut dire qu'un tel système laissé sans contrôle reste à peu près dans le même état ou oscille entre deux états. Bien sûr cela dépend de la métrique utilisée pour comparer les formes de l'organisation, et également du seuil ε qui définit l'équivalence entre deux formes (deux formes A et B sont dites ε-équivalentes si $d(A,B) < \varepsilon$).

L'analyse des cas qui présentent deux formes uniquement révèle qu'en général il y a une forme prédominante. Si l'on considère les deux formes, indexées 1 et 2, la distribution du rapport $\frac{f_1}{f_2}$ exprimant la proportion de conformation du système en ces deux formes, est illustrée sur la figure 6.3.10. Cette figure révèle que dans ces systèmes « binaires », une forme est prédominante, comme l'indique le pic d'abscisse $0,7-0,9$. On observe également une population de cas, plus restreinte, qui a plutôt tendance à osciller entre les deux formes (abscisse $0,3-0,5$).

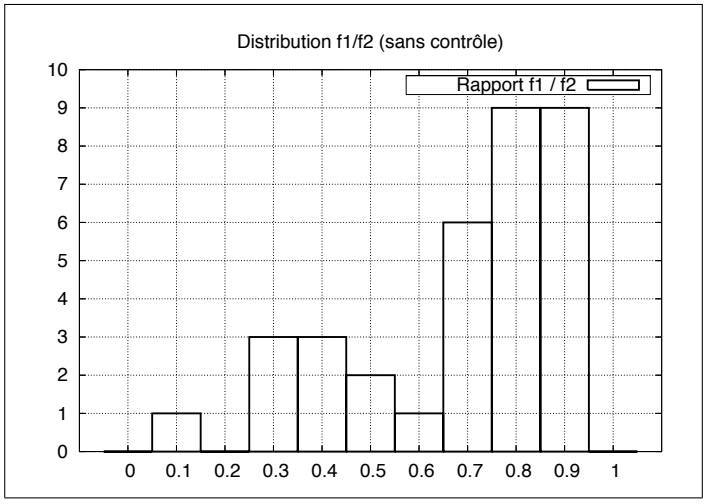


Figure 6.3.10 – *Distribution* $\frac{f_1}{f_2}$

Les figures 6.3.11 et 6.3.12 représentent les distributions des abscisses moyennes et coordonnées moyenne de l'objet déplacé au cours du temps (toujours selon des paramétrages différents du système, et sans contrôle).

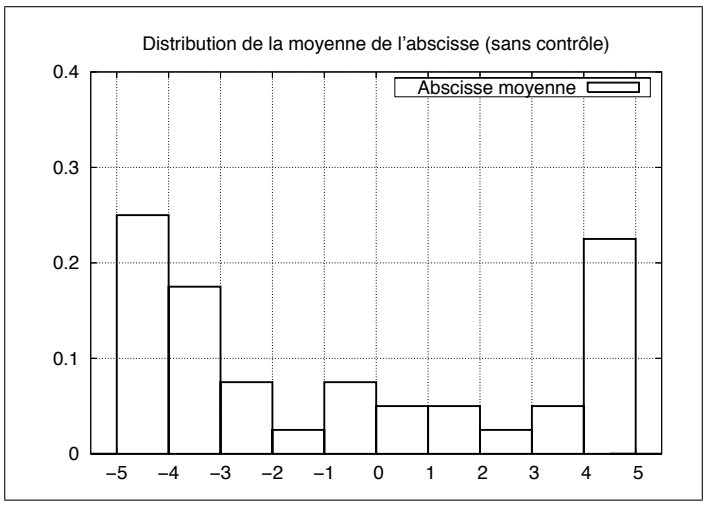


Figure 6.3.11 – *Distribution de l'abscisse moyenne sans contrôle.*

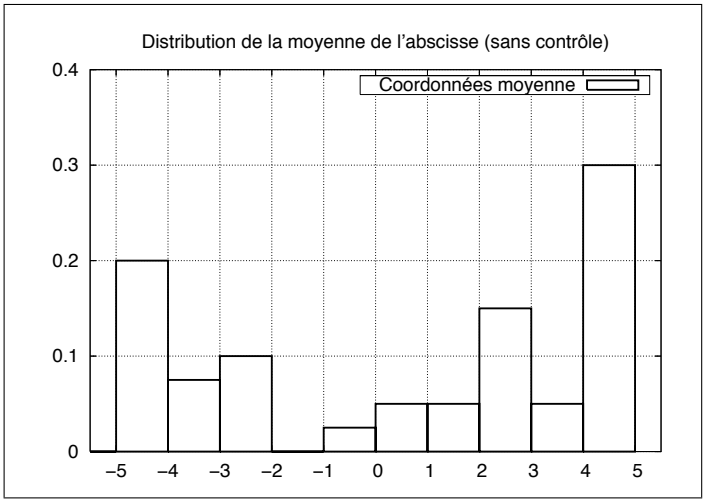


Figure 6.3.12 – *Distribution de la coordonnée moyenne sans contrôle.*

On peut constater que la position moyenne correspond à ce que l'on avait observé sur la figure représentant les traces de l'objet (figure 6.3.7) : l'objet est poussé vers les régions des bords du carré virtuel $(-5,-5)-(5,5)$. Le petit pic en $0,0$ correspond au début de la simulation. Il y a un temps de latence, correspondant à l'éveil des agents.

Avec contrôle

À présent, nous allons nous intéresser au comportement du système lorsqu'on le contrôle. La tache donnée au système est d'essayer de déplacer l'objet dans une direction donnée, par exemple vers les x positifs (vers la droite si l'on veut).

L'apprentissage des formes commence à l'itération 100 et le contrôle effectif commence à l'itération 250 sur 1000 cycles de simulation.

Les figures 6.3.13 et 6.3.14 illustrent les distributions des abscisses moyennes et coordonnées moyennes de l'objet poussé au cours du temps, cette fois-ci avec contrôle de la part du module d'analyse).

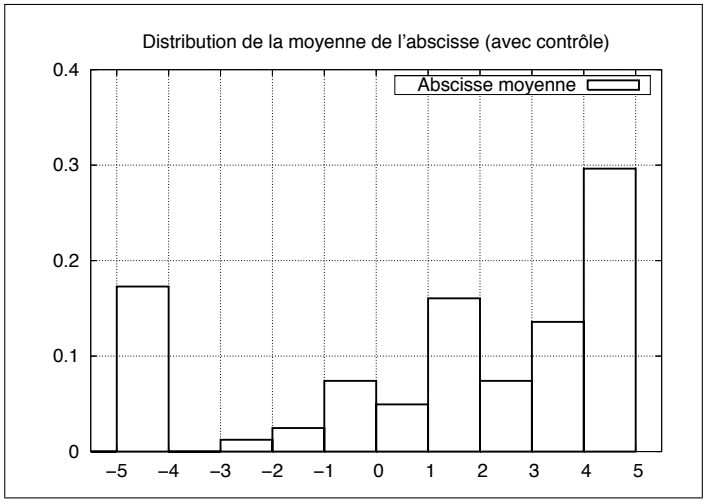


Figure 6.3.13 – *Distribution de la moyenne de l'abscisse avec contrôle. La moyenne est globalement décalée vers les x positifs.*

On observe, sur la figure 6.3.13, un net décalage vers la droite de la moyenne de l'abscisse. Cependant on constate qu'il y a toujours un ensemble de cas (un peu moins de 20%) dans lesquels l'objet est très souvent à gauche (abscisses $-5; -4$). On constate aussi que la partie centrale $(-1; 1)$ n'est pas très affectée, cela correspond à la période de transition du système ou le module d'apprentissage est inactif. L'apprentissage n'a lieu qu'à partir de l'itération 100 pour ne par encombrer l'ensemble d'apprentissage de formes correspondant à des états du système trop particuliers

La figure 6.3.14 montre que la position moyenne de l'objet selon l'axe des Y est peu affectée.

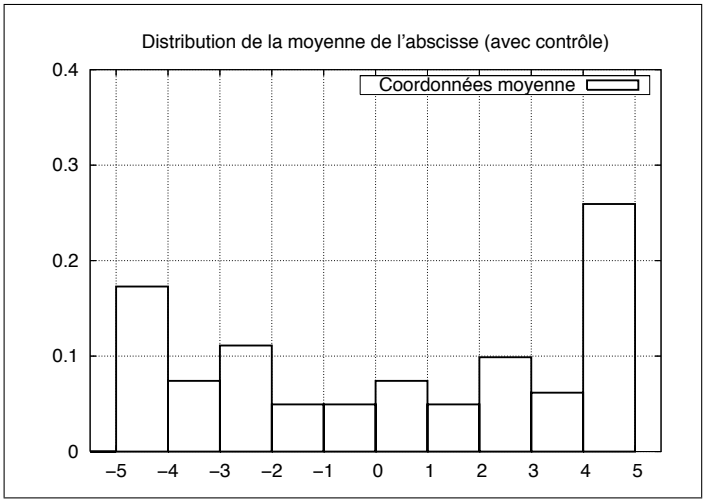


Figure 6.3.14 – *Distribution de la coordonnée moyenne avec contrôle. Cette distribution est très peu affectée par le contrôle, contrairement à celle de l'abscisse.*

Les figures 6.3.15 à 6.3.17 illustrent le comportement typique du système. Sur ces figures la position moyenne sur l'abscisse du système est reportée dans le deux cas :

- le cas sans contrôle, en ligne pointillée ;
- le cas avec contrôle, en points foncés.

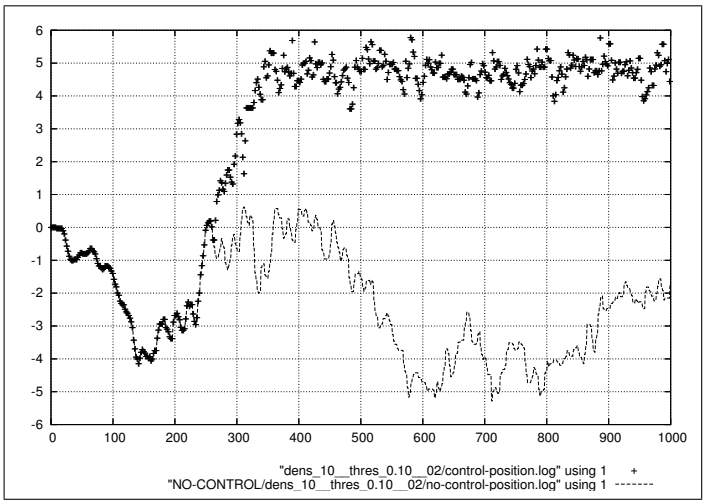


Figure 6.3.15 – *Abscisse moyenne avec contrôle :* 2.82975.

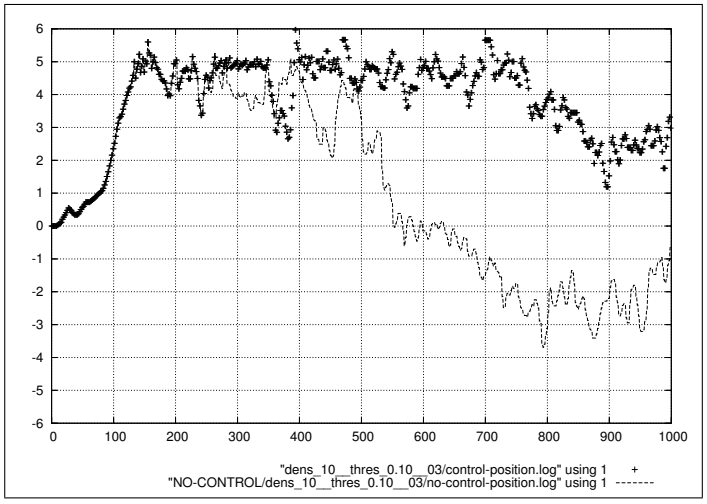


Figure 6.3.16 – *Le contrôle maintient l'objet à droite (abscisse moyenne* 3.79792*), alors que sans contrôle, l'objet est repoussé à gauche.*

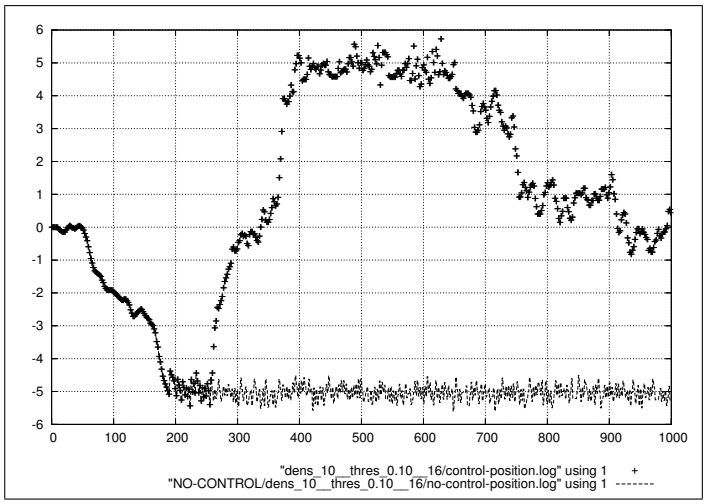


Figure 6.3.17 – *Dans ce cas, sans contrôle l'objet est vite poussé à gauche. Avec le mécanisme de contrôle, l'objet est repoussé à droite au bout d'un moment.*

Les figures 6.3.18 et 6.3.19 montrent les cas « extrêmes » du comportement du système :

– le système se comporte convenablement de façon spontanée (sans contrôle), et le contrôle est redondant ;

– le système se comporte mal et le contrôle est inefficace.

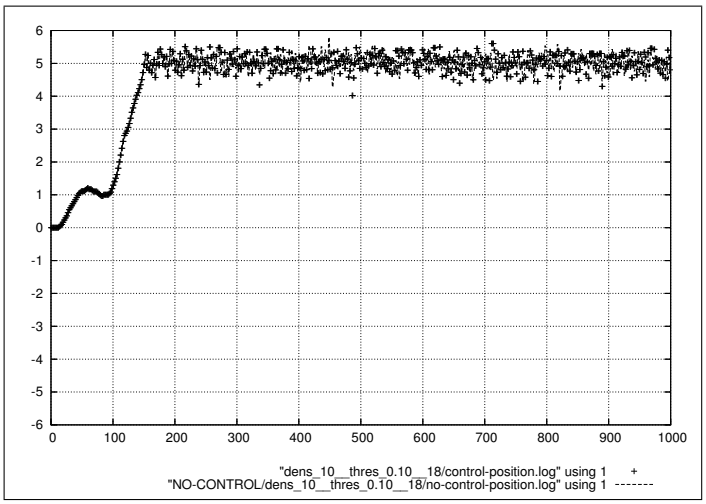


Figure 6.3.18 – *Dans certains cas le système pousse l'objet de façon spontanée à droite (abscisse moyenne avec contrôle* 4.50795*).*

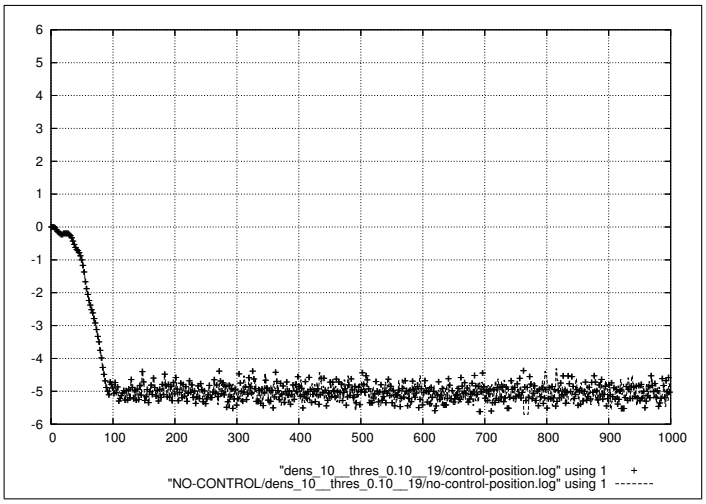


Figure 6.3.19 – *Dans certains cas, le système ne parvient pas du tout à contrôler (abscisse moyenne avec contrôle :*-4.65675*)*

La figure 6.3.20 illustre la distribution du nombres de formes dans les graphes d'évolution obtenus avec contrôle. La première observation est que le nombre de

formes apprises est plus grand (jusqu'à 38 formes), alors que sans contrôle, le maximum était de 11. Cette augmentation du nombre de conformations prises par l'organisation aspectuelle est due au fait qu'elle est contrainte seulement à certains moments par l'analyse. Rappelons que lorsque des agents aspectuels simples reçoivent un ordre de changement de rôles, les agents ne pouvant changer de rôle se mettent en veille. Lorsqu'ils sont en veille, ils « disparaissent » momentanément de l'espace de morphologie : il ne sont pas pris en compte par la morphologie. En quelque sorte, la variation du nombre d'agents pris en compte dans l'espace morphologique est un degré de liberté supplémentaire dans la variation des formes, ce qui explique cette augmentation du nombre de formes.

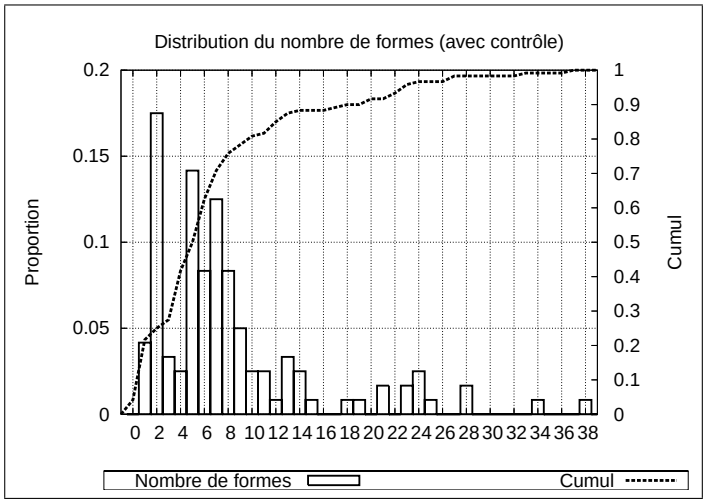


Figure 6.3.20 – *Distributions du nombre de formes apprises pour un ensemble de paramétrage du système (avec contrôle).*

6.3.2 Comparaison de formes

Les figures 6.3.21 et 6.3.22 illustrent deux graphes d'évolution extraits avec des paramètres de comparaison de formes respectifs de $0,1$ et $0,2$. Nous nommerons ces cas $C1$ et $C2$ par la suite. Les performances respectives des ces deux systèmes sont reportés sur la figure 6.3.23. Le système dans le cas $C1$ a un meilleur contrôle que dans le cas $C2$.

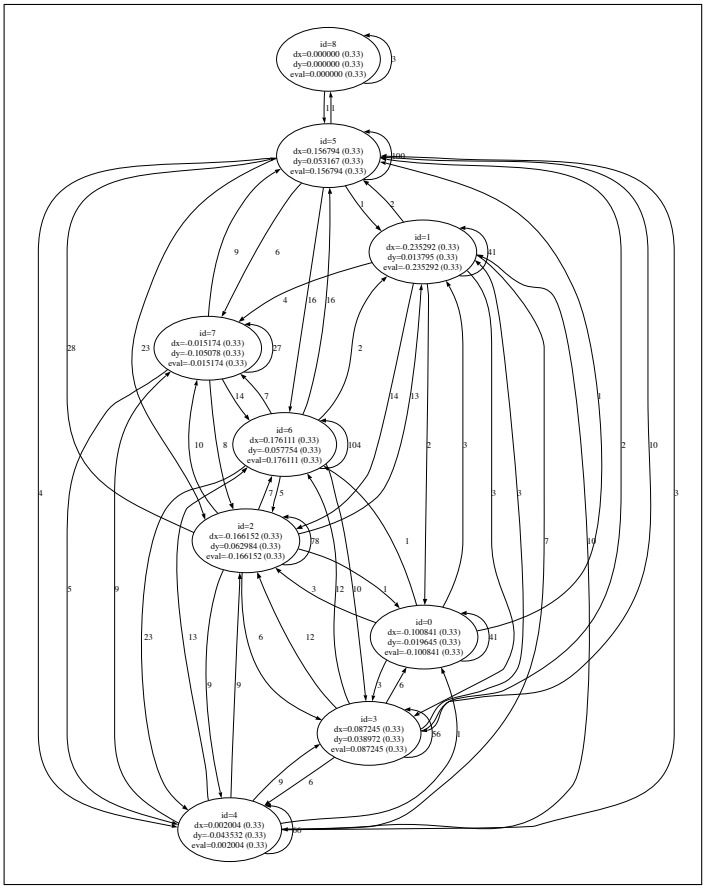


Figure 6.3.21 – *Graphe d'évolution extrait avec une valeur de comparaison de formes de* $0,1$ *(C1).*

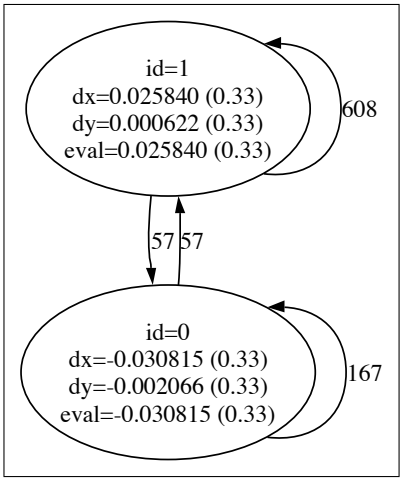


Figure 6.3.22 – *Graphe d'évolution extrait avec une valeur de comparaison de formes de* $0,2$ *(*$C2$*).*

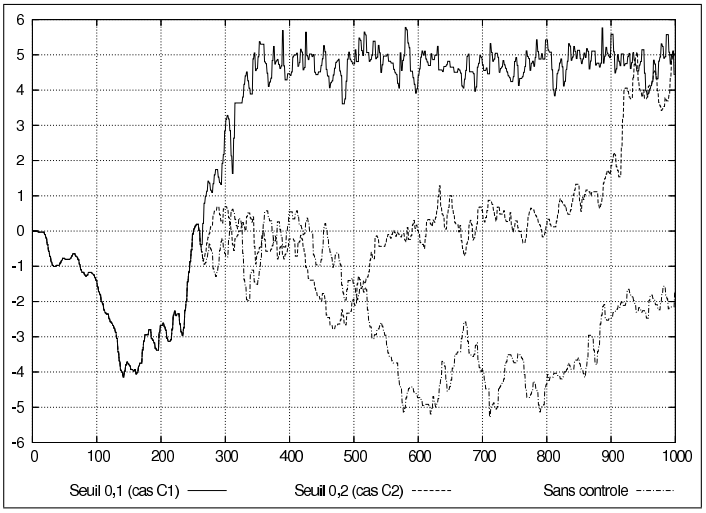


Figure 6.3.23 – *Évolution de l'abscisse pour des seuils de comparaison différents (*$0,1$ *et* $0,2$*), et sans contrôle*

La première constatation est que le graphe d'évolution dans le cas $C1$ comporte plus de formes que dans le cas $C2$, neuf formes contre deux formes. Ceci est naturellement dû au seuil de comparaison des formes. Par ailleurs, ayant davantage de formes, le systèmes présente plus de transitions entre les différentes

formes. Notons cependant que toutes les transitions entre toutes les formes ne sont pas présentes.

6.4 Remarques et extensions

Dans ce chapitre, nous avons présenté un deuxième exemple d'application du contrôle par l'analyse morphologique d'une organisation d'agents.

Cet exemple nous a permis de mettre en évidence certains aspects du contrôle morphologique appliqué à un système plus complexe. Nous pouvons observer que le choix de la métrique, ainsi que la méthode de comparaison des formes, jouent un rôle central dans le contrôle.

Les extensions envisageables de cette application sont l'approfondissement de l'étude de l'influence des différents paramètres entrant en jeu dans le calcul de la morphologie et l'apprentissage effectué. Par ailleurs, il serait intéressant de prendre en compte la dynamique, en étudiant les mécanismes de transitions du modèle construit, afin de mieux comprendre les lois d'évolution d'un tel système.

Troisième partie

Perspectives

Chapitre 7

Émotions artificielles

Dans ce chapitre nous nous intéressons aux émotions artificielles. Nous commencerons par spécifier la notion d'émotion en biologie en exposant rapidement les différents définitions et modèles de mécanismes d'émotions (section 7.1). Ensuite nous verrons que, depuis quelques temps, la notion d'émotion a suscité de l'intérêt dans la communauté de l'intelligence artificielle ; nous passerons en revue certains travaux liés à cette notion (section 7.2). Finalement nous proposerons des pistes pour un modèle de générateur d'émotions fondé sur un système multi-agents en suivant une voie strictement constructiviste (section 7.3).

7.1 Émotions en biologie

7.1.1 Généralités

Les émotions sont vues en biologie comme des réponses chimiques et neuronales qui possèdent un rôle régulateur, celui de maintenir l'organisme (le corps qui en est le théâtre) en vie. La fonction biologique de l'émotion est double. D'une part elle consiste en la production d'une réaction spécifique à la situation inductrice et d'autre part elle procède à la régulation de l'état interne de l'organisme de telle sorte qu'il puisse être prêt pour la réaction spécifique. Les émotions sont induites par le traitement de certaines situations (ou objets) par le dispositif sen-

soriel de l'organisme, ou bien à partir de certains objets ou situations évoqués par la représentation mentale de l'organisme, à partir de sa mémoire.

L'émotion n'est pas à confondre avec l'affectivité. Tout changement de l'environnement créé, pour un organisme vivant, une réaction d'approche ou d'évitement. C'est lorsqu'il y a une disproportion entre les exigences d'une situation donnée et ses possibilité de réponse que l'émotion naît. Même si l'émotion apparaît comme une profonde désorganisation du comportement empêchant le sujet de s'adapter aux exigences immédiates de la situation présente, c'est un moyen d'adaptation relativement primitif mais parfois très efficace. L'émotion devient alors une source de motivation pour le sujet qui l'éprouve. Par ailleurs, l'émotion assurerait une importante fonction de communication entre individus.

Les émotions ne sont pas non plus à confondre avec les réflexes ni la douleur. Les réflexes sont constitués de comportements simples. La douleur correspond à la conséquence d'un dysfonctionnement local dans un tissu vivant.

Types d'émotions

On peut distinguer trois grandes classes d'émotions (bien que les frontières entre ces trois catégories ne soient pas nettes) :

- émotions primaires ou universelles : bonheur, tristesse, peur, colère, surprise, dégoût ;
- émotions secondaires ou sociales : embarras, jalousie, culpabilité, orgueil ;
- émotions d'arrière-plan :bien-être, malaise, calme, tension.

Les émotions d'arrière-plan sont induites de façon interne, qui peuvent être provoquées par les processus de régulation biologique, mais également des conflits mentaux lorsqu'ils suscitent une satisfaction ou une inhibition. Ces émotions ne sont pas facilement détectables. Les humeurs résultent de la prolongation de certaines émotions (comme le stress).

Les émotions primaires (ou de base) sont plus faciles à définir. Ce sont les émotions les plus étudiées en neurobiologie.

Les émotions sociales comme la sympathie, l'embarras, la honte, la culpabilité, l'orgueil, l'envie, la gratitude admiration, l'indignation et le mépris, ne sont pas uniquement présentes chez les humains, mais sont présentes aussi dans des sociétés animales.

Les humains ont la possibilité d'être conscients de la relation qui existe entre certains objets et certaines émotions. Nous pouvons délibérément lutter pour contrôler nos émotions, du moins dans une certaines mesure [Dam00].

Bien que le terme d'émotion, possède diverses définitions selon les auteurs, il est possible de retenir ces trois aspects :

- expérience subjective ;
- activation/changement physiologique ;
- changement de comportement.

7.1.2 Théories sur les émotions

Nous allons rapidement passer en revue quelques différentes théoriques historiques sur les émotions en biologie, puis nous présenterons la notion d'émotion sous une forme plus actuelle.

Théorie de James-Lange

William James et Carl Lange [Jam90] proposèrent, vers 1880, une théorie selon laquelle un stimuli qui provoque une émotion engendre un changement de l'état physiologique (voir figure 7.1.1). L'émotion est l'interprétation cognitive de ce changement physiologique. En d'autres termes, il s'agirait de produits secondaires des réponses automatiques physiologiques et comportementales. L'implication est que l'on pourrait alors contrôler l'émotion en contrôlant le comportement : pleurer rend triste, sourire rend heureux...

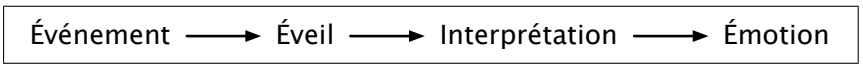


Figure 7.1.1 – *Théorie des émotions de James-Lange.*

Cette théorie est contre-intuitive, et elle fut largement critiquée. Un argument fort en défaveur de cette théorie est le suivant : des animaux ou des humains désafférés (c'est-à-dire dont les nerfs entre les viscères et le cerveau sont sectionnés) présentent toujours des émotions. On pourrait encore dire que même si un animal présente les signes d'émotions, qu'est-ce qui nous dit qu'il en fait l'expérience ?

Théorie de Cannon-Bard

Selon la théorie de Canon et Bard [Can27], proposée en 1927, les changements physiologiques de l'organisme et l'expérience subjective se dérouleraient en même temps. Le thalamus joue le rôle important de relais sensoriel : il peut envoyer des signaux au cortex pour interprétation et simultanément envoyer des signaux d'activation au reste du corps par le biais de l'hypothalamus. Ces deux phénomènes seraient donc indépendants : l'expérience subjective ne causerait pas et ne dépendrait pas des changements physiologiques (voir figure 7.1.2). Cela pourrait alors expliquer pourquoi des sujets désafférés pourrait faire l'expérience d'émotions. Cannon avait sans doute tort de donner au thalamus un rôle si important.

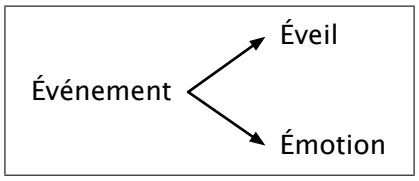


Figure 7.1.2 – *Théorie des émotions de Cannon-Bard.*

Théorie de Schachter-Singer

Dans cette théorie, l'émotion dépend de deux facteurs : un éveil, une activation non spécifique du système nerveux et ensuite une interprétation cognitive de cet éveil (voir figure 7.1.3).

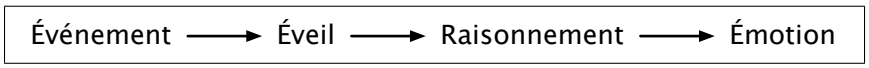


Figure 7.1.3 – *Théorie des émotions de Schachter-Singer.*

L'émotion serait de fait le résultat d'une interaction entre deux composantes, d'une part, un éveil non spécifique dû à l'activation du système nerveux sympathique et, d'autre part, une cognition résultant de ce que l'on attribue la cause de cet éveil à un élément de la situation provocatrice.En d'autres termes, un individu fait l'expérience d'un éveil de l'activation physiologique et tente d'interpréter cet évènement. Si cette interprétation n'a pas de contenu émotif, alors l'individu n'aura pas d'émotion. En revanche, dans le cas contraire, l'individu fera l'expérience d'émotion(s). Le degré de l'éveil aura aussi une influence sur le degré de l'émotion.

Théorie de Lazarus

Dans la théorie de Lazarus [Laz91], une pensée doit avoir lieu avant toute émotion ou éveil physiologique. En d'autres termes, une personne songe à sa situation avant de pouvoir ressentir une émotion...

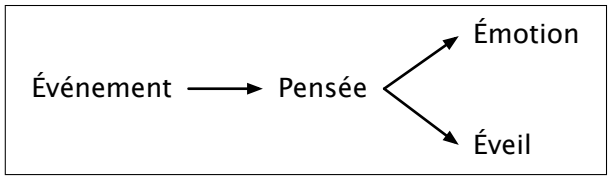
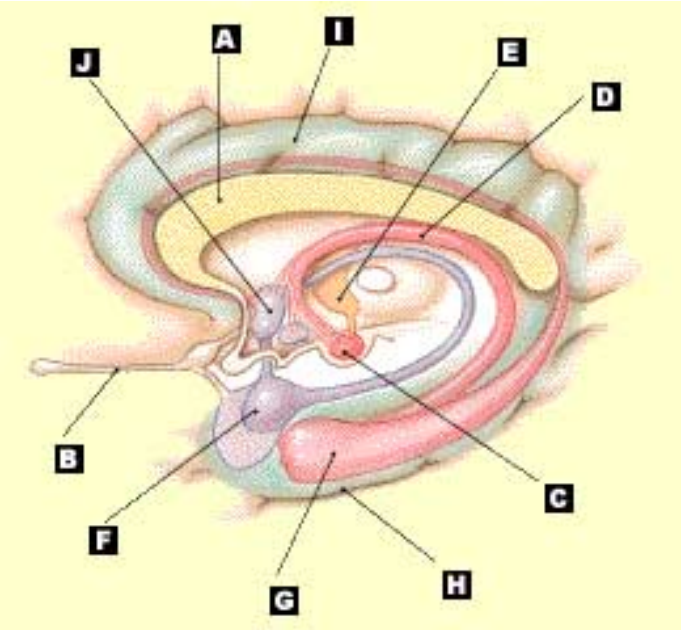


Figure 7.1.4 – *Théorie des émotions de Lazarus.*

7.1.3 Mécanismes des émotions

Nous adopterons davantage le modèle et les définitions proposés par A. Damasio [Dam00].

Les régions du cerveau identifiées à ce jour comme sites de déclenchement des émotions (ce ne sont pas les seuls, mais il s'agit de ceux que l'on connaît le mieux) : les sites sous-corticaux : comprenant l'amygdale, l'hypothalamus et le télencéphale basal ; et les sites corticaux (partie du lobe frontal, la région cingulaire antérieure, la région préfrontale ventro-médiane).



- A : Corps calleux
- B : Voie olfactive
- C : Corps mammilaires
- D : Fornix
- E : Noyaux antérieur du thalamus
- F : Amygdale
- G : Hippocampe
- H : Gyrus parahippocampique
- I : Circonvolution cingulaire
- J : Noyaux hypothalamiques

Figure 7.1.5 – *Aggrandissement sur la partie centrale d'une coupe du cerveau.*

Certains sites sont plus ou moins spécifiques à certaines émotions. Par exemple la tristesse fait intervenir le cortex préfrontal ventro-médian, l'hypothalamus et le tronc cérébral. En revanche, la colère et la peur ne font pas intervenir le cortex préfrontal ni l'hypothalamus (voir figure 7.1.5).

L'amygdale est essentielle à notre capacité de ressentir et de percevoir chez les autres certaines émotions. C'est le cas de la peur et de toutes les modifications corporelles qu'elle entraîne. Chez certains patients qui ont dû subir une intervention chirurgicale au cerveau, on a pu stimuler l'amygdale directement et recueillir leurs impressions. L'expérience subjective la plus commune décrite est celle de danger imminent et de peur. Les très rares patients dont uniquement l'amygdale a été détruite (lors d'accident cérébraux vasculaires par exemple) reconnaissent toutes les expressions émotionnelles sur les visages sauf celle de la peur. L'amygdale semble en fait moduler toutes nos réactions à des événements qui ont une grande importance pour notre survie. Ceux qui nous avertissent d'un danger imminent

sont donc des stimuli très importants pour l'amygdale, mais également ceux qui signalent la présence de nourriture, de partenaires sexuels, de rivaux, d'enfants en détresse, etc...

Le lobe frontal, en particulier la région préfrontale, participe au déclenchement d'émotions sociales. L'équipe de A. Damasio a montré que des lésions du lobe frontal altèrent l'aptitude à avoir des émotions lorsque le stimulus émotionnellement compétent est de nature sociale et lorsque la réponse correcte est une émotion sociale comme l'embarras, la culpabilité et le désespoir [Dam00].

La mise en évidence de la participation des différentes aires ou structures dans les différentes émotions a été établie par l'observation du comportement de personnes dont les différents sites sont atteints ou détruits, étude complétée par l'observation par neuro-imagerie fonctionnelle du cerveau.

7.1.4 Caractérisation des émotions

L'expression des émotions est étroitement liée au système nerveux végétatif. Elle met en jeu tous les centres cérébraux qui contrôlent les neurones préganglionnaires du tronc cérébral et de la moelle. Le système végétatif (appelé encore autonome, ou viscéral) contrôle les fonctions involontaires qu'exercent les fibres musculaires lisses, fibres musculaires cardiaques et les glandes. Le système nerveux végétatif comporte deux principales divisions et une troisième semi-indépendante :

- le sous-système sympathique, qui mobilise les ressources de l'organisme pour faire face aux différentes situations qui se présentent ;
- le sous-système para-sympathique, qui lui, en revanche, intervient dans les états de calme relatif pour restaurer l'énergie dépensée ;
- et le troisième : système nerveux entérique, qui correspond à l'innervation spécialisée du tube digestif.

De façon un peu simplifiée, nous pouvons dire que le rôle du système parasympathique est généralement opposé à celui du système sympathique (par exemple : contraction de la pupille, ralentissement du rythme cardiaque...). Il existe cependant certains organes qui ne sont qu'innervés par le système sympathique, par exemple : la médullosurrénale, les muscles piloérecteurs et la plupart des vais-

seaux sanguins.

Cette régulation nerveuse permanente de la dépense et du renouvellement des ressources de l'organisme contribue dans une large mesure à l'équilibre global des fonctions organiques que l'on appelle l'homéostasie. Alors que les centres principaux qui contrôlent l'activité motrice somatique sont représentés par le cortex moteur primaire et secondaires des lobes frontaux, les centres qui contrôlent le système nerveux autonome sont majoritairement situés dans l'hypothalamus et dans les circuits complexes, mal délimités. Donnons quelques exemple de l'activation du système végétatif sympathique qui se traduit par :

- la dilatation des pupilles et rétraction des paupières (plus de lumière, meilleure vision) ;
- contraction des vaisseaux sanguins pour faire affluer le sang vers les autres muscles ;
- les poils se dressent ;
- augmentation de la fréquence cardiaque ;
- les fonctions de digestions diminuent en activité ;
- libération d'adrénaline par la médullosurrénale ;
- libération d'insuline par le pancréas ;
- ...

En quelques mots, l'activation du système végétatif tend à faire diminuer l'activité des fonctions inappropriées dans certaines conditions et favorisent celles orientées vers la survie de l'organisme. Cependant il faut nuancer... Le système sympathique ne répond pas par tout ou rien, le but de son activité tonique est de maintenir ses organes cibles (pupille, cœur, bronches, organes digestifs...) à un niveau opérationnel approprié à toutes les situations.

Lors du déroulement d'une émotion, certaines régions du cerveau envoient des commandes à d'autres régions et presque aussi partout dans le corps. Ces commandes sont envoyées par deux routes : par la circulation sanguine (molécules chimiques) et par les voies neuronales (signaux électrochimiques agissant sur voies neuronales-muscles, autres neurones, autres organes). Ces commandes ont pour effet de provoquer un changement global dans l'état de l'organisme. Les effets possibles sont par exemple :

- les organes changent de fonction ;
- les muscles lisses de vaisseaux sanguins ou visage se meuvent...
- le cerveau est aussi affecté.

En résumé, le processus d'une émotion peut être décrit en trois étapes [Dam00] (voir figure 7.1.6) :

1. une induction émotionnelle à partir d'un objet (traité par l'appareillage sensoriel
2. des signaux consécutifs au traitement activent des sites neuronaux (réponses innées mais modulées par l'expérience) ;
3. les sites d'inductions de l'émotion déclenchent des signaux vers d'autres sites neuronaux du cerveau ou dans le corps.

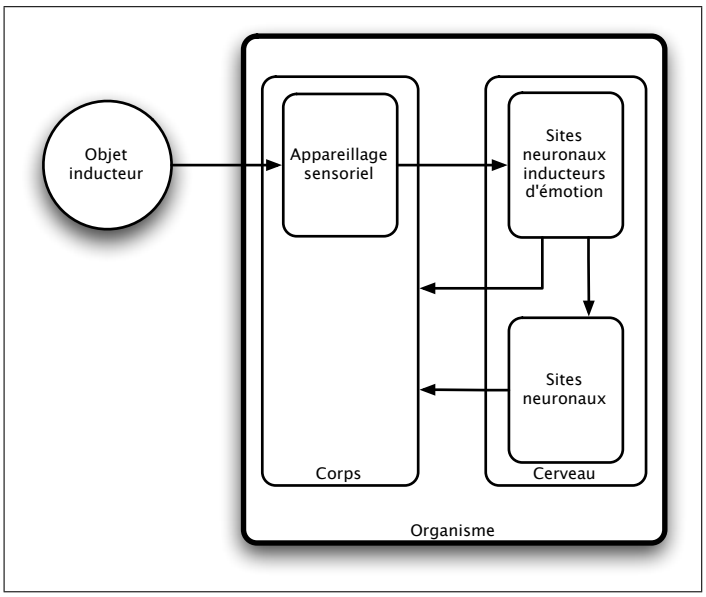


Figure 7.1.6 – *Schéma très simplifié illustrant le mécanisme des émotions.*

7.1.5 Quelques remarques

Les émotions font parties des dispositifs bio-régulateurs dont les organismes vivant sont équipés pour vivre et survivre. Les émotions sont universelles, on les retrouve tout aussi bien chez des individus de cultures différentes ou même à travers des espèces différentes, selon des degrés plus ou moins élaborés. Même les organismes simples éprouvent des émotions. Par exemple, Damasio [Dam00] cite l'exemple de la mouche Drosophila melanogaster qui aurait des émotions,

mais cela ne veut pas dire qu'elle les ressent. Le système de contrôle est très sophistiqué (les mouches ont aussi besoin de sommeil). Cette mouche n'a pas de moelle épinière. Damasio mentionne également l'exemple de l'escargot de mer Aplysia californica. Cette animal ne possède pas de moelle épinière, mais des réactions d'émotion (la pression sanguine augmente, le pouls est plus fort) peuvent être constatées. Cependant, il n'a sans doute pas de sentiment (l'animal ne perçoit pas consciemment ses émotions). Ces réactions d'émotions ne sont pas produites suite à la suite d'une délibération, il s'agit de mécanismes automatique. Il n'est pas approprié de les appeler réflexes. Ce sont des collections de réponses réflexes, assez élaborées, et coordonnées dans l'ensemble.

Les émotions sont caractérisables sur le plan biologique de la manière suivante [Dam00] :

1. les émotions sont un ensemble de réponses chimiques et neuronales. Elles possèdent un rôle régulateur, leur rôle est d'aider à maintenir l'organisme (le corps) en vie ;
2. même si l'apprentissage et la culture modifient l'interprétation des émotions, celles-ci sont des processus biologiques déterminés, qui dépendent de dispositifs cérébraux établis de façon innée, mis en place en place au terme d'une longue histoire évolutionnaire ;
3. les dispositifs qui produisent des émotions occupent un ensemble relativement restreint de régions sous-corticales ;
4. tous ces dispositifs peuvent être mis en route automatiquement sans délibération consciente ;
5. toutes les émotions utilisent le corps comme théâtre. Les émotions affectent également de nombreux circuits cérébraux (changements dans le paysage corporel et le paysage cérébral).

7.2 Émotions artificielles

7.2.1 Motivation

Selon les biologistes, certains aspects de l'intelligence humaine, notamment la capacité à prendre des décisions dans des environnements dynamiques et indéterminés, dépendraient des émotions. On peut constater les insuffisances de l'intelligence artificielle classique à fournir des méthodes heuristiques aussi puissantes que les capacités mentales humaines. On peut se tourner alors vers la vie artificielle qui postule que les facultés cognitives sont fondées sur la capacité à maintenir son intégrité dans des environnements complexes et dynamiques [And03]. Nous venons de voir qu'un des mécanismes fondamentaux de l'adaptation est l'émotion. M. Minsky et A. Sloman avaient suggéré d'entreprendre des recherches sur les émotions afin d'en doter des entités artificielles, en vue de rendre ces systèmes plus adaptatifs, plus autonomes, plus aptes à faire face aux réalités du monde. Cependant ce n'est que récemment que l'intelligence artificielle s'est réellement intéressé aux émotions artificielles, notamment avec les ouvrages de Damasio et de Goleman. Les travaux de M. Minsky [Min03] ont d'ailleurs évolués pour inclure les résultats issus des nouvelles recherches, notamment ceux de Damasio [Dam94, Dam00, Dam04].

Le comportement d'un système basé sur l'émotion présente des avantages par rapport à un système à base de connaissance, et il a aussi des inconvénients. Si le domaine d'action du système est fini, totalement connaissable et connu, un comportement optimisé est évidemment le meilleur. Aucune émotion n'apporte d'avantage décisif. En revanche, si le système a un domaine d'action ouvert, où des situations nouvelles peuvent se présenter, un comportement émotionnel doit permettre la sauvegarde du système en générant des plans d'actions « inspirés », adaptés continûment et définis rapidement, sans souci du meilleur choix calculable. Une émotion est un réducteur adapté de la recherche de solution adéquate.

Plusieurs cas sont à envisager :

- si le système est isolé, une émotion permet de sauvegarder son intégrité : peur devant les attaques virales (annihilation volontaire de processus), plaisir d'offrir plus de résultats que demandé, ce qui peut intéresser le demandeur ..
- si le système appartient à un groupe (groupe de robots, groupe ouvert d'ap-

plications logicielle), l'émotion est le seul garant de l'intégrité du groupe. Seules les émotions permettent à des ensembles de systèmes (ou de robots) de former des groupes cohérents et de rester « spatialement » des groupes, par besoin qu'a chacun de ressentir avec besoin la proximité des autres.

Nous pouvons distinguer trois grandes motivations, d'axes différents, pour l'étude des émotions sous le paradigme de la vie artificielle :

- dans le « paraître », communication (interface homme-machine), par exemple [Bat97] ;
- dans la modélisation des émotions biologiques ;
- et dans l'apprentissage et adaptativité de systèmes artificiels, par exemple ([VPF98, Vel97].

7.2.2 Quelques modèles

Nous allons présenter quelques travaux sur les émotions dans les systèmes artificiels, essentiellement orientés dans l'amélioration de l'apprentissage du système où elles interviennent.

Travaux de Ventura

Le modèle de Ventura [VPF98] est fondé sur la théorie du marqueur somatique issue des travaux de Damasio [Dam94]. La théorie du marqueur somatique stipule que certains événements vécus sont mémorisés en association avec la réponse normale de l'organisme. En d'autres termes : certaines images perçues lors d'un événement sont marquées et interviennent dans la représentation courante de l'organisme.

Le modèle de Ventura repose sur une double représentation :

- une image cognitive, destinée à la reconnaissance de motifs. Elle est suffisamment riche pour permettre une bonne reconstruction du stimulus original ;

– une image perceptuelle, plus simple, plus réduite, consistant en traits essentiels qui sont davantage significatifs pour l'entité.

Ceci se traduit, dans le modèle, par la distinction de deux couches, une couche dite cognitive et une couche dite perceptuelle. Ventura insite sur le fait que même si une couche est appelée « perceptive », les deux couches concernent effectivement le stimuli perçu.

La couche perceptuelle est composée de l'image perceptuelle mais également d'un vecteur que Ventura appelle le « vecteur de désirabilité » (« Desirability vector »). Chaque composante de ce vecteur représente en fait une évaluation simple d'un aspect du stimulus. Chaque composante peut être soit active, soit neutre. Si une composante se trouve dans un état neutre, cela indique que l'évaluation n'a pas été effectuée pour le critère associé. En revanche, une composant dite « activée » signifie que le stimulus a provoqué une évaluation particulière. Ventura donne l'exemple de la peur : un stimulus néfaste pour l'entité pourra activer la composante « Peur » du vecteur de désirabilité, qui déclenchera un comportement de peur associé.

Dans la présentation de son modèle, Ventura emploie le terme agent. En effet, Ventura adopte la définition « cognitive » du mot agent. Il considère en fait son système comme un agent qui éprouve des émotions artificielles. Ce point de vue diffère radicalement du notre, où l'agent est une petite entité logicielle, mais nous utiliserons dans ce paragraphe la terminologie de l'auteur.

Le fonctionnement du modèle de Ventura est le suivant. En réponse à un stimulus externe, les deux couches perceptuelles et cognitives traitent l'information en parallèle. Au niveau de la couche perceptuelle, il y a une relation directe entre le stimulus et le vecteur de désirabilité. Cette fonction est partiellement prédéfinie à la création de l'agent. Néanmoins cette fonction est adaptative. Cette mémoire fait penser à la mémoire phylogénétique. Le traitement dans la couche cognitive consiste à reconnaître des formes dans le stimulus, en les comparant avec la mémoire de la couche cognitive. Cette mémoire, à l'opposé de la mémoire perceptuelle, contient des associations vécues. Ces associations consistent en l'image cognitive, le vecteur de désirabilité correspondant et de l'image perceptuelle. Les actions effectuées pas l'entité en réponse au stimulus proviennent essentiellement du vecteur de désirabilité. Lorsque l'agent effectue une action sur l'environnement, ce dernier peut changer. Ce fait peut être perçu par l'entité comme un retour de stimulus (« feedback »). Le nouveau stimulus permet de dire à l'agent le résultat de l'action effectuée, et de faire de l'apprentissage. L'apprentissage, dans le

modèle de Ventura, peut se faire au niveau de la couche perceptuelle, en modifiant la sensibilité de la fonction à certains aspects du stimulus, et également dans le couche cognitive, en marquant une ou plusieurs images cognitives avec le vecteur de désirabilité et les actions qui ont conduits à un changement dans l'environnement.

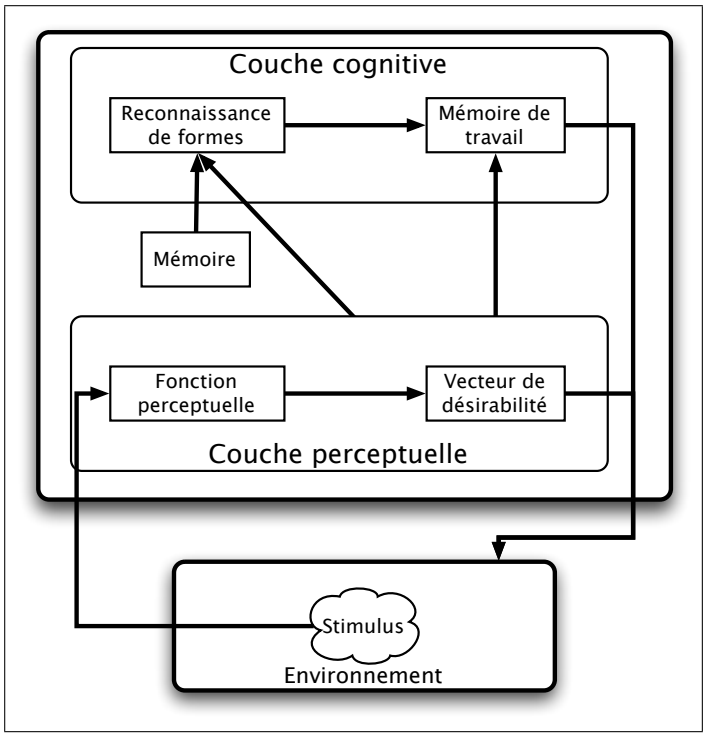


Figure 7.2.1 – *Modèle de Ventura*

Ventura propose trois implantations de son modèle [dMV00] :

- « damasio », un système relativement simple développé pour expérimenter essentiellement le mécanisme du marquage somatique ;
- « faces », qui est une amélioration du système « damasio » pouvant traiter des stimuli plus complexes ;
- et « decks », un système réalisé afin de reproduire l'expérience du jeu de cartes décrit pas Damasio dans [Dam94].

L'auteur stipule que dans ces trois cas, le modèle a permis aux différents systèmes de pouvoir gérer les situations imprévues de façon efficace, sans pour autant

passer trop de temps à traiter l'aspect cognitif du stimulus.

Modèle de Velásquez

Velásquez [Vel97] propose un modèle également fondé sur les travaux de Damasio. Le modèle est résumé schématiquement sur la figure 7.2.2. Les éléments du modèle sont appelés des modules. Le système perceptif obtient des informations à partir de l'environnement et il met à disposition les stimuli au module de comportement et au module d'émotions. Ces modules reçoivent également du module de motivation des signaux d'erreurs. Le module d'émotions évalue le contenu émotionnel des stimuli et biaise ainsi le module de comportement de même que le module de perception. Le module de comportement produit et effectue les actions moteur appropriées. Précisons que les modules consistent en plusieurs sous-systèmes, cependant les sous-systèmes correspondants ne sont pas réunis de façon monolithique dans des modules.

Tout comme Ventura, Velásquez, utilise la définition cognitive du terme agent. Dans son modèle, Velásquez considère que son système est un agent qui a des émotions artificielles.

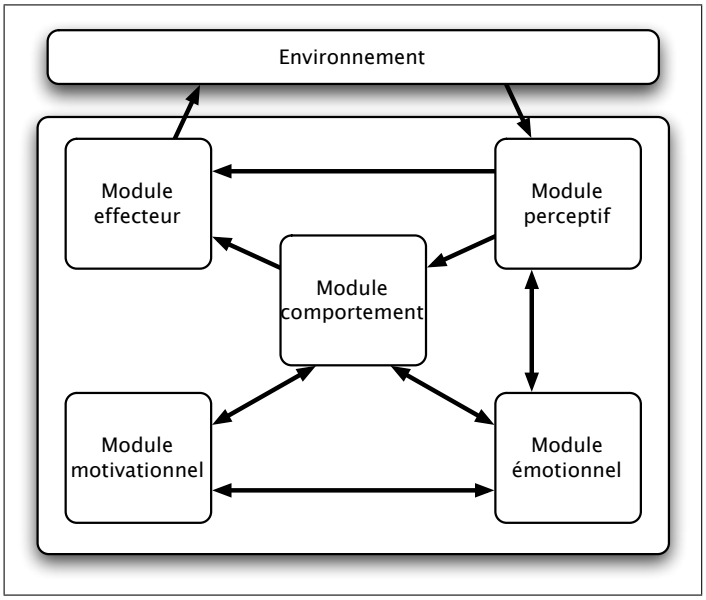


Figure 7.2.2 – *Le modèle de Velásquez*

L'entité de base dans le modèle de Velásquez est une unité computationnelle non-linéaire. Un sous-système peut être vu comme étant constitué d'entrées, un système d'évaluation, et de sorties. La fonction d'activation d'un système peut-être écrite sous la forme :

$$A_i = f(\sum_k R_{ki} W_{ki})$$

où :

- A_i est l'activation du système i ;
- R_{ki} est la valeur du déclencheur k ;
- W_{ki} est son poids associé ;
- l'indice k varie sur tous les déclencheur du système i.

Ceci n'est sans rappeler les réseaux de neurones artificiels.

Les systèmes motivationnels représentent les pulsions du système qui poussent celui-ci à s'engager dans l'action. Par exemple, la motivation « faim » devra aider à contrôler les sous-systèmes qui interviennent dans la prise de nourriture par le système. Velasquez souligne la différence entre le module motivation et le module émotionnel. Dans son modèle c'est le module émotionnel qui constitue les principales motivations du système. Par ailleurs le module motivationel exploite ce module pour altérer le module de comportement. Les systèmes motivationels sont en fait des systèmes de contrôle qui maintiennent une variable régulée dans un domaine précis. Cette variable est mesurée à l'aide d'une sorte de capteur de l'agent et cette valeur est comparée à une valeur de référence. S'il y a un écart, un signal d'erreur est généré. Ce signal d'erreur est donnée au système de motivation adéquat et ce signal peut être combiné à d'autres signaux d'erreur. L'activation d'un système de motivation ce fait selon l'équation donnée précédemment.

Les système émotionnels représentent différentes familles de réponses affectives telles que la panique, la peur. Une classe d'émotion est caractérisée par des mécanismes propres et par des caractéristiques telles que la similitude des événements qui précèdent l'expression de celle-ci.

Velasquez considère quatre classes de déclencheurs d'émotions :

- les neutres, qui incluent les effets des neurotransmetteurs, de la température du cerveau, les hormones... ;
- les sensorimoteurs, qui prennent en compte les aspects sensorimoteurs des émotions, tels que les expressions faciales, la posture... ;
- les motivationnels, qui prennent en considération les effets des motivations qui provoquent des émotions (la soif, la peur, la faim, la joie...) ;
- et les cognitifs, qui sont les processus cognitifs qui potentiellement provoquent des émotions, tels que l'évaluation, la mémorisation...

Dans son modèle, les trois premiers déclencheurs sont considérés comme naturels, ou innés et le quatrième appris avec l'expérience. L'activation des systèmes de motivation est un peu différente de celle des autres systèmes. Elle peut s'écrire sous la forme suivante :

$$A_i(t) = f(\Phi(A_i(t-1)) + \sum_k R_{ki} W_{ki} + \sum_l \mu_{li} A_i(t))$$

Cette équation, un peu plus compliquée, intègre une partie excitatrice et inhibitrice ainsi qu'une prise en compte d'un effet de demie-vie.

Le système de comportement est également constitué de systèmes possédant chacun leurs déclencheurs, reliés aux autres systèmes adéquats. Ces déclencheurs réifient en fait certaines choses ou certains faits tel que, par exemple « la batterie est rechargée ». Les déclencheurs du module de comportement peuvent interférer entre eux, soit en s'inhibant ou en s'amplifiant mutuellement. L'activation d'un déclencheur est également spécifiée par une équation non-linéaire.

Velasquez a réalisé plusieurs instanciation de son modèle [Vel97] :

- « Simón the Toddler » (simulé) ;
- « Virtual Yuppy » (simulé)
- et Yuppy qui correspond au portage de Virtual Yuppy sur robot réel.

Modèle de Gadanho

Gadanho [GH01b, GH01a, GH] propose un modèle exploitant la notion d'émotion, fondé également sur les travaux de Damasio, pour l'apprentissage en robotique. Les émotions que le modèle prend en compte sont au nombre de quatre :

- le bonheur ;
- la tristesse ;
- la peur ;
- la colère.

Gadanho justifie ce choix en mentionnant que ce sont les émotions les plus répandues universellement. Dans son modèle (voir figure 7.2.3), l'intensité de chacune des émotions est déterminée en fonction des sentiments internes du robot : la douleur, la température, la faim, l'odeur, la chaleur, la proximité... Gadanho définit les émotions par un ensemble de dépendances constantes des sentiments et par un biais. Les valeurs de ces dépendances sont fixées. Le choix des valeurs est important pour générer les émotions adéquates selon l'état du robot.

Gadanho spécifie un seuil d'activation pour les émotions à partir duquel l'émotion agit effectivement sur le système de contrôle du robot par le biais du système hormonal. Le système hormonal permet de moduler la valeur issue du système de sensation. La valeur d'une hormone peut être suffisamment forte (positivement ou négativement) pour complètement biaiser les sensations réelles du robots. Ce système d'hormones permet aux émotions d'entrer en compétition pour le contrôle du comportement du robot (figure 7.2.4). Le module de comportement est affecté par les valeurs des émotions qui permettent un apprentissage, rendant le contrôle adaptatif.

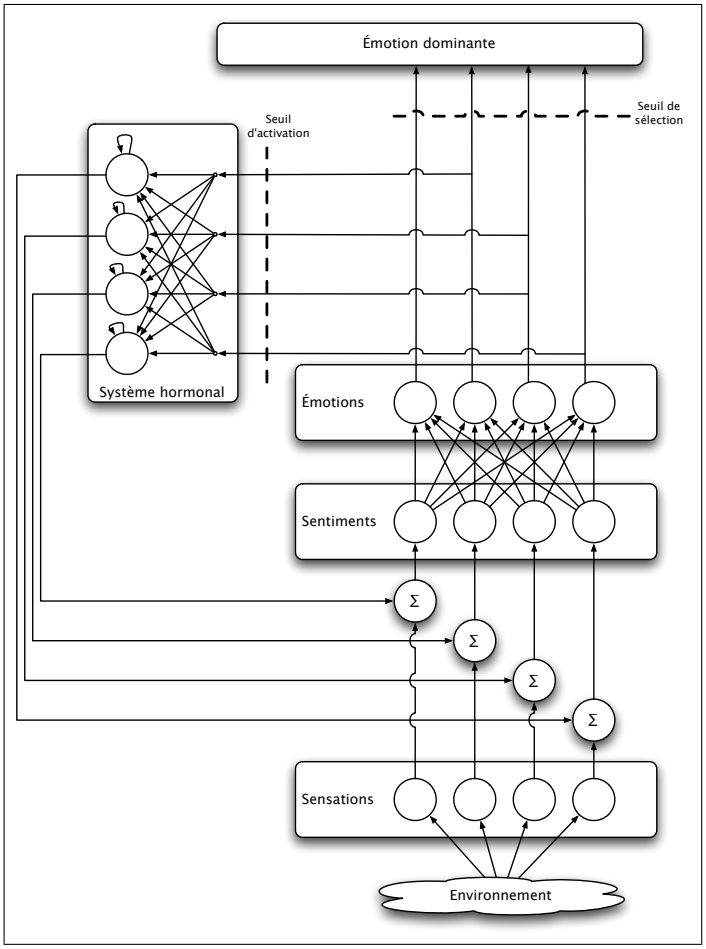


Figure 7.2.3 – *Modèle d'émotions artificielles de Gadanho.*

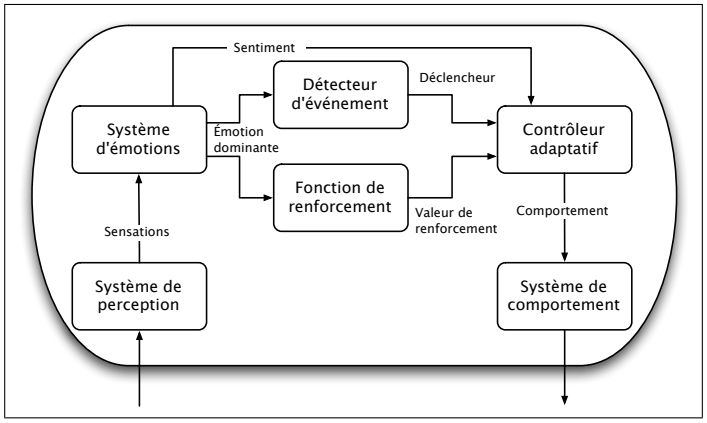


Figure 7.2.4 – *Système de contrôle du robot proposé par Gadanho.*

Le modèle de Gadanho a été appliqué et testé sur un robot Khepera simulé. Le but du robot est de se nourrir en se cognant dans des lumières qu'il pouvait percevoir vaguement. Plus le robot roulait vite, plus son énergie diminuait rapidement. Hormis ce but, le robot devait également éviter de se cogner de façon accidentelle contre les murs ou rester sur place trop longtemps. Les conclusions que donne l'auteur est que le robot avait en général un comportement plausible, étant donnée la situation dans laquelle il se trouvait.

Gadanho proposa après une extension de son modèle qui incorpore une couche cognitive d'interprétation en vue de rendre le système de contrôle du robot plus performant [Gad03].

Travaux de A. Sloman

A. Sloman [Slo01b] met en garde contre l'usage abusif du terme émotion. Il soulève les problèmes de dialogue qu'entraîne le manque de vocabulaire commun. Pour cela, il propose un environnement de travail, CogAff, en espérant fournir un modèle qui pourra servir de base commune pour dialoguer et tenter de lever les ambiguïtés. Il propose également et une plate-forme de développement d'agents : SIM_AGENT [Bir].

SIM_AGENT est une boite à outils fournissant plusieurs paradigmes de programmation :

- programmation orientée objet ;
- programmation procédurale ;
- programmation à base de listes ;
- programmation à base de règles ;
- réseaux de neurones.

La boite à outil permet la construction d'un ensemble d'agents où chaque agent possède une architecture multi-composant, où les différents composants opèrent en parallèle, utilisant différents mécanismes. Ces différents modules, d'un point de vue implantation, peuvent être de natures différentes (systèmes à base de règles, réseaux de neurones...) et se retrouver à différents niveaux (réactif, délibératif et niveau méta ; voir figure 7.2.5). L'évaluation des différents systèmes à base de règles peut introduire des conflits dans l'évaluation de ces règles. Cependant dans SIM_AGENT des règles concurrentes, ou indépendantes, peuvent être déclenchées en même temps et en parallèle pour les niveaux réactifs. Pour les niveaux délibératifs, il est possible de mettre en place un mécanisme de choix. Même si SIM_AGENT propose une librairie multi-paradigme intéressante, cette librairie ne propose pas vraiment de méthode d'analyse ni de contrôle.

Une fois l'architecture définie, Sloman décrit certains phénomènes dans le système supporté par cette architecture, et en spécifie certains comme pouvant s'apparenter à des émotions [Slo01a]. Différentes formes d'émotions apparaissent à certains niveaux de l'architecture. Par exemple, pour le niveau réactif, il donne l'exemple de la distinction qui peut être faite pour le système opérant normalement et lorsqu'une menace plane sur l'organisme. Lorsqu'une menace est détectée, le système peut être redirigé, le comportement normal du système est court-circuité par un mécanisme d'alerte. Ce genre de mécanisme peut être considéré comme des « proto-émotions ». Sloman propose des pistes pour l'étude et l'implantation des émotions à travers l'analyse de phénomènes qui peuvent se produire au sein de son architecture. Cependant Sloman reste relativement général dans ses descriptions. Par exemple, on ne voit pas très bien comment la couche méta opère sur les autres couches, quelle représentation des sous-couches elle utilise.

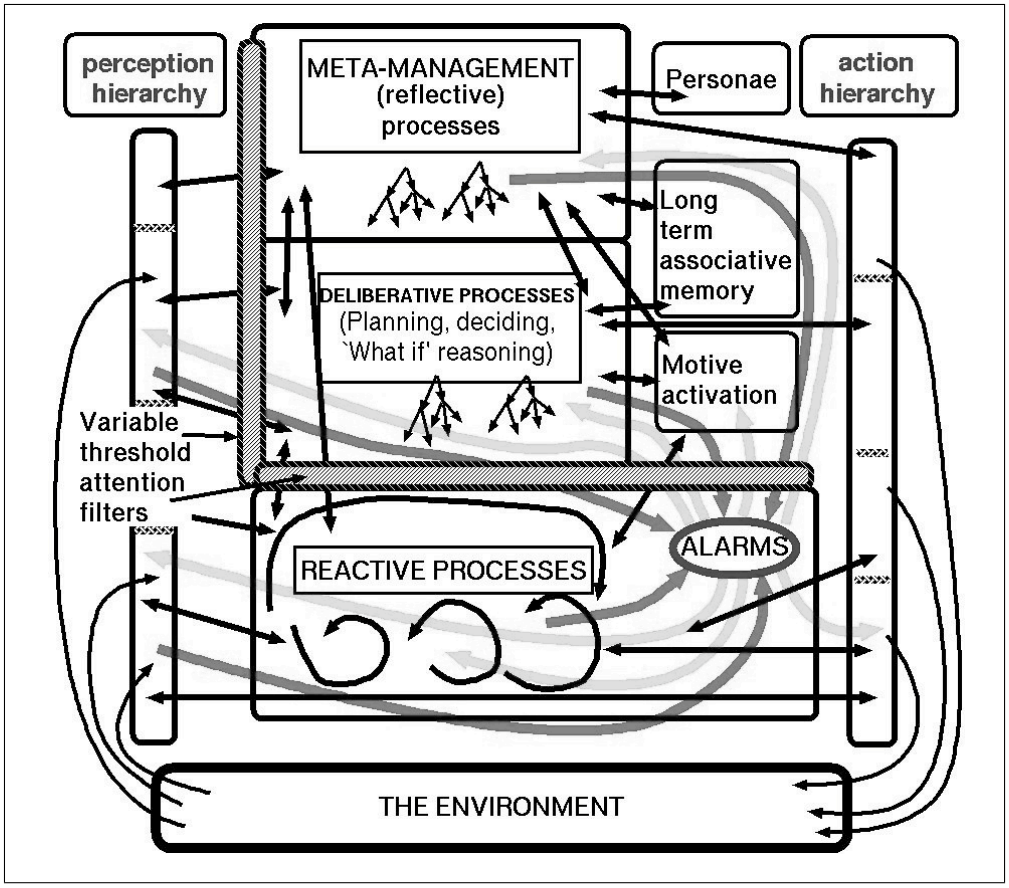


Figure 7.2.5 – *Modèle H-CogAff de Sloman (images tirée de [Slo04]).*

Les remarques de Sloman sur l'émotion et les modèles simples (« shallow models ») sont pertinentes. Sa boite à outils, Sim_Agent, intègre bien différents paradigmes de programmation, mais elle ne fournit pas réellement de mécanismes propres à leur observation ou à leur pilotage. C'est à l'utilisateur de les spécifier et de les construire à l'aide de la boîte à outils.

Travaux de M. Minsky

M. Minsky propose, dans son ouvrage « Emotion machine » [Min03], encore sous forme de brouillon mais disponible librement, des pistes pour une architecture de machine qui pourrait avoir des émotions au même titre que les humains.

Enfin, il va plus loin que de proposer un modèle pour les émotions, il envisage des pistes pour construire une machine pensante. Son ouvrage est intéressant dans la démarche, mais il ne propose pas directement de modèle opérant : il propose au lecteur de poursuivre le parcours de réflexion qui l'amène à son modèle.

Quelques remarques

Nous avons vu que l'émotion suscite, comme la notion d'agent, beaucoup d'intérêt en intelligence artificielle. Sans pour autant être exhaustif, ce bref exposé permet d'avoir une idée générale sur les types de travaux menés dans ce cadre. Il y a des propositions relativement réductrices, utilisant la notion d'émotion biologique comme une justification de choix d'architecture et sans vraiment mettre en place de systèmes analogues à ceux du vivant, et, par contraste, il y a des suggestions de modèles plus élaborés qui tentent de se rapprocher des mécanismes biologiques. Nous avons vu que beaucoup de recherches s'inspire des travaux de Damasio, très souvent cité. Néanmoins, même si Damasio insiste sur le fait que les émotions sont des processus complexes, bien souvent les travaux en vie artificielle conduisent à des modèles très simplifiés. Les émotions reposent sur un système, comme nous l'avons vu dans la section 7.1 et sont des phénomènes complexes qui se déroulent dans le temps. Beaucoup des travaux en intelligence artificielle prennent en compte ce premier aspect, mais souvent réduisent le second à une variation d'une valeur de variables intitulée « peur » ou « joie »... D'autres, tels ceux de Minsky ou de Sloman par exemple, tentent de prendre en compte cet aspect de phénomène complexe qui se déroule dans le temps pour proposer des modèles.

7.3 Pistes proposées

7.3.1 Motivation

L'un des objets de cette thèse était de proposer un système de contrôle pour un robot en s'appuyant sur les paradigmes de la vie artificielle (corporéité, émotions...) et des systèmes multi-agents. Néanmoins le problème de la construction d'un système multi-agent présupposait une méthode d'agentification, une méthode d'analyse et de contrôle de tels systèmes. À présent nous proposons des

pistes pour l'élaboration d'un modèle d'émotions artificielles fondé sur le paradigme multi-agents.

Le bref exposé précédent sur les émotions en biologie nous a permis de mettre évidence le coté systémique de ces phénomènes. Les émotions dans les organismes vivants ont lieu par le biais d'un mécanisme assez bien décrit par les biologistes et est relativement bien précisé de nos jours.

L'idée de pouvoir piloter un robot à l'aide d'une architecture spécifique pourrait profiter de cette notion d'émotion, comme beaucoup d'autres auteurs le suggèrent. Néanmoins, nous avons vu que dans beaucoup de travaux, la notion d'émotion est bien souvent réduite à la variation d'une variable particulière. Tout l'aspect systémique, et surtout l'aspect dynamique du mécanisme, n'est pas pris en compte. Nous proposons un modèle qui prendra en compte cet aspect.

Les deux idées fondamentales de notre approche sont :

- la prise en compte du fait que la représentation mentale est répartie dans le cerveau et consiste en une activation globale. Une idée n'est pas localisée précisément dans le cerveau. Il a été observé que certaines facultés ayant « disparu » suite à une lésion cérébrale, pouvaient « réapparaître » par la suite [Dam00]. Ceci sera repris dans notre proposition par une répartition de la représentation distribuée et localisée dans les agents ;
- par ailleurs, la prise en compte de l'ensemble des agents et de leur contrôle se fera par le biais d'une analyse géométrique, le contrôle morphologique.

Notre proposition se distingue des autres approches, telles que celles de Ventura, Ventura et Gadanho, qui, même si elles s'inspirent des découvertes en neurobiologie, proposent des modèles réductionnistes. Notre approche se rapproche davantage de celles de Sloman, en ce qu'il propose de considérer l'émotion artificielle effectivement comme un processus. Néanmoins, la première différence est qu'il utilise la définition d'agent cognitif, pour désigner son système en entier comme étant un agent, alors que nous nous situons dans le contexte multi-agent pour construire un système. Au delà de cette première distinction qui peut paraître ne porter que sur le vocabulaire, la seconde différence est que Sloman ne propose pas de méthode d'observation ou de contrôle explicite dans son modèle. Nous suggérons d'adopter une approche morphologique de l'étude de l'organisation.

7.3.2 Propositions

Le système proposé par Alain Cardon pour faire se conduire intentionnellement un robot dans un environnement quelconque est basé sur l'architecture suivante :

- les organisations d'un système multi-agent, dit de représentation, génèrent en continu une représentation mentale que se fait le robot lui-même, situé dans son environnement ;
- un système multi-agent, dit générateur des émotions, joue le rôle de système limbique : il engage à la production d'émotions en faisant se modifier le comportement du système de représentation ;
- le résultat au sens organisationnel du système de représentation engage à l'activation des effecteurs du robot.

Une émotion est donc traitée au niveau de la représentation courante du robot à propos de ce que doit être son intérêt, son attention et son action. C'est une modification de l'état mental du robot, qui entraîne un comportement typique. Un tel type d'émotion doit permettre sa manipulation interne par le système de représentation, et doit donc pouvoir être considérée comme une émotion artificielle. Nous ne décrirons que le système générateur des émotions dans cette thèse. Le développement du système complet fait l'objet d'un projet industriel.

7.3.3 Entité dynamique fondamentale dans l'organisation d'agents : l'oscillateur calculable

Nous allons nous concentrer sur la notion de structure permettant de réaliser effectivement une émotion. Nous ne retiendrons absolument pas l'approche réductrice qui pose que l'émotion est la valeur d'une variable e pouvant prendre des valeurs sur $[0, 1]$ par exemple.

La présentation biologique de la génération d'émotions est très claire quant à l'existence de domaines jouant des rôles fonctionnels précis les uns par rapport aux autres dans une certaine partie du cerveau. Mais il s'agit d'une description fonctionnelle avec des entrées et des sorties moléculaires entre des parties, des retours de boucle positifs ou négatifs. Rien, avec un tel modèle, ne permet de comprendre ce que peut être la finesse et la complexité d'une émotion si ce n'est

par un gradient de production de molécules, rien ne permet de comprendre ce qu'elle est vraiment ni en quoi elle peut saturer un individu un moment au point qu'il ne ressent que cette émotion précise, en étant de plus contraint à la mémoriser et l'interpréter pour la connaître et en jouer. Nous devrons résoudre ces problèmes au niveau d'une nouvelle architecture, en inventant de nouveaux systèmes.

L'émotion artificielle sera initiée dans un système spécifique inclus dans un système plus global. Nous reprenons l'architecture neuronale isolant le cerveau du corps et distinguant bien le système méso-limbique, initiateur des émotions, de la partie corticale où se construit la représentation des choses.

Nous posons que le système générant des émotions est constitué :

- de deux couches interfaces d'entrée et de sortie ;
- d'une organisation interne d'entités proactives, cette organisation générant des boucles typiques d'activités ;
- d'une organisation pulsionnelle fixant les tendances ;

Nous allons fonder l'architecture du système produisant des émotions sur le composant de base suivant, essentiellement dynamique : un oscillateur calculable. La définition proposée par A. Cardon [CC05] est la suivante :

> Oscillateur calculable
> Il s'agit d'une organisation d'agents opérant dans le système de représentation, dont l'activation collective forme un cycle ayant une forme et une période typiques et relativement stables pendant une durée significative. Cet oscillateur est nécessairement déclenché par au moins un agent de pulsion.

Les caractères représentatifs d'un oscillateur calculable sont les suivants (voir figure 7.3.1) :

1. il y a un système avec des entrées et des sorties et il y a, distinct mais fortement couplé, un système de représentation, d'interprétation des entrées et des sorties. Ce système de représentation est composé d'agents (qui seront appelés agents aspectuels structurants) ;
2. il y a, à chaque instant de l'activation des agents du système de représentation, la description géométrique des actions de ces agents dans les dimen-

sions de l'espace morphologique, c'est-à-dire qu'il y a de disponible un nuage de points d'une certaine forme géométrique, comme un graphe ;

3. ce nuage de points (ou ce graphe) a aussi une caractérisation sémantique issue de la sémantique portée par les rôles des agents.

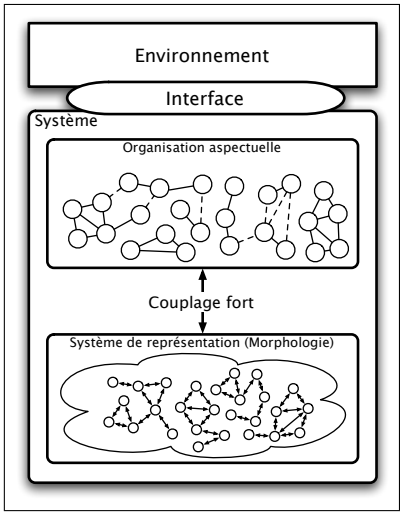


Figure 7.3.1 – *Oscillateur calculable*

Ainsi, un tel oscillateur est en fait un point de l'espace morphologique (de $(\mathbb{R}^9 \times S)^n$, si l'on retient ces dimensions, où n est le nombre d'agents distingués du système de représentation où $\mathbb{R}^9$ est l'espace de représentation de dimension 9 et S est l'espace sémantique des rôles des agents. On va s'intéresser à la trajectoire de ces points (de ces nuages de points ou de ces graphes) produisant une forme dynamique typique qui sera ou non un cycle limite activé avec un certain rythme. La notion de cycle représente, au niveau de la sémantique S, celle de chaîne causale.

Interprétation d'un oscillateur

Une autre question porte sur l'interprétation de chaque point de l'espace morphologique ($(\mathbb{R}^9 \times S)^n$) et sur la trajectoire formée par ces points. Disons que ce point est lui-même un polyèdre coloré par des qualités sémantiques. Il peut y avoir une qualité dominante ou plusieurs qualités distinctes (il y a n agents ayants chacun un rôle dominant). Le polyèdre peut être régulier ou non et avoir ou non une

forme typique. La trajectoire est une déformation régulière, une extension ou non d'une forme typique.

Nous dirons que la trajectoire dans l'espace morphologique formant un certain oscillateur exprime une émotion artificielle si :

- elle a la forme typique d'un graphe cyclique, non permanent mais se déformant en conservant sa forme limite (attractive et typique) ;
- elle peut systématiquement se produire pour des classes typiques d'entrées du système interface d'entrée-sortie ;
- elle peut s'auto-entretenir pendant une durée cohérente avec les actions du système ;
- elle altère le comportement rationnel du système (altération des agents aspectuels).

Une interprétation est donc la caractérisation d'un cas de fonctionnement systématique d'un ou de plusieurs oscillateurs qui prennent le pas sur le fonctionnement de groupes d'entités standards assurant des rôles fonctionnels dans le système et ne générant pas de forme claire.

Il sera nécessaire de définir une architecture très adaptée pour permettre à la fois de voir émerger un oscillateur et d'obtenir son interprétation, une architecture semblable fonctionnellement au système méso-limbique.

Le système développé sera systématiquement architecturé pour produire un ensemble de tels oscillateurs, formant des groupes, des organisations d'oscillateurs se couplant et se déployant. Une telle organisation doit se contrôler pour passer d'un état formant une structure coopérative globale (un graphe complet de communications et d'actions) à un état où un certain processus d'activation cyclique rapide de certains ensembles d'agents s'est dégagé (un sous-graphe émergent de forme typique et en activation de plus en plus rapide et bien synchronisée d'agents). Ces oscillateurs seront observés dans l'espace morphologique étendu par la sémantique des rôles, évidemment différent d'un espace réduit aux simples processus. Il est à noter que ce type de comportement est tout à fait semblable aux formes et effets des attracteurs dans les systèmes dynamiques non linéaires, où les formes des attracteurs dans les espaces de phases sont typiques d'un certain comportement chaotique des systèmes complexes décrits par des équations différentielles partielles.

Le système sera finalement formé d'un ensemble bien architecturé d'agents logiciels légers, des agents fonctionnant pour produire des oscillateurs, permettant leur renforcement global et local, et aussi leur inhibition et arrêt.

Illustration

Nous allons nous baser sur l'application précédente du chapitre 6 pour illustrer la notion d'oscillateur calculable et de cycle limite. L'application, rappelons-le, consistait en un système multi-agent agent qui peut déplacer des objets dans un environnement. Selon la structure de l'organisation d'agents, le comportement global du système est différent, sa fonction est différente. Nous allons donner deux exemples où le système oscillera entre deux formes et, ensuite, entre quatre formes.

Dans ces expériences, nous faisons apprendre au système à déplacer l'objet dans certaines directions et ensuite nous forcerons le système à osciller entre les différents états correspondant aux différents types de mouvement.

Le but de ces exemples est d'illustrer qu'il est possible d'utiliser le contrôle morphologique pour faire tendre le système vers un équilibre non statique. Nous verrons, comme nous l'avions indiqué en introduction de la thèse, que le contrôle ne sera pas forcément total.

Première expérience Dans un premier temps nous allons contraindre le système à se conformer sous deux formes et de façon périodique. Ces deux formes correspondent d'un point de vue fonctionnel du système à un déplacement vers le haut à gauche et à un déplacement vers le bas à droite des objets. Les figures de 7.3.2 à 7.3.5 résument l'expérience. Elles illustrent le déplacement de l'objet quand l'organisation est contrainte à osciller entre deux formes correspondant aux deux fonctions de déplacement de l'objet apprises par le système. Le basculement entre les deux formes est, dans cette simulation, provoqué périodiquement. Le module d'analyse change le choix de la forme de façon cyclique tous les 1000 pas d'itération. Cela se traduit par un déplacement cyclique de l'objet dans l'objet comme illustré sur les figures suivantes.

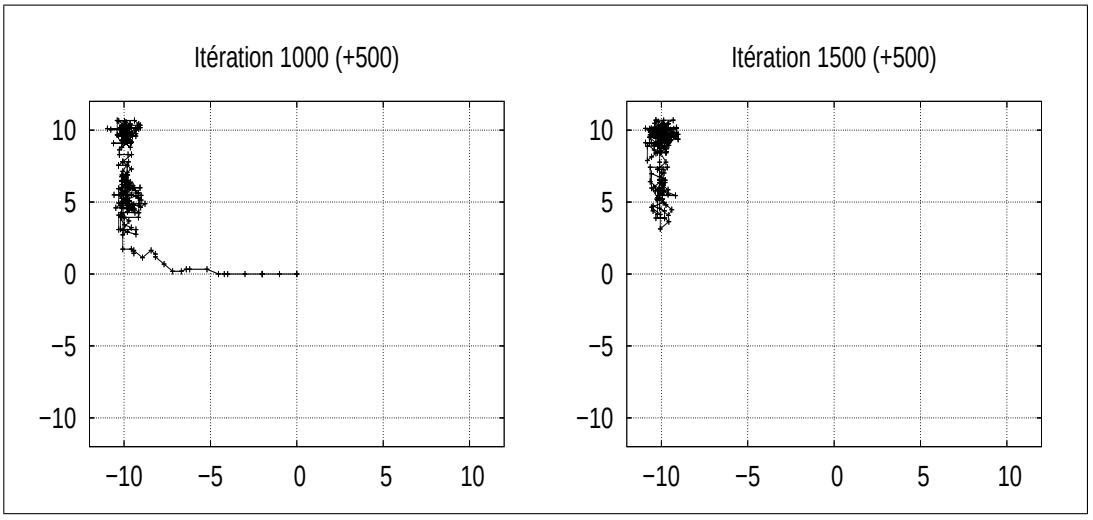


Figure 7.3.2 – *L'expérience commence avec l'objet situé à l'origine, en* $(0,0)$. *Le module d'analyse contraint l'organisation sous la première forme. Le système réalise la fonction de déplacement de l'objet vers le haut à gauche (Itérations 1000-2000).*

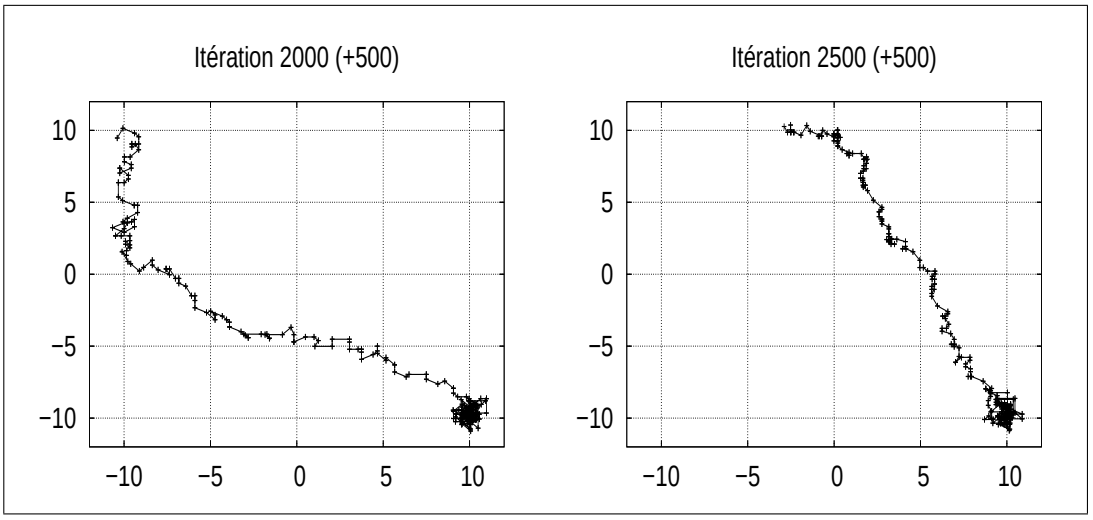


Figure 7.3.3 – *À l'itération 2000, le contrôle contraint l'organisation d'agent à adopter la seconde forme. L'objet se déplace alors vers le coin en bas à droite. Le système devrait rester dans cette deuxième conformation, cependant l'organisation se déforme et l'objet est déplacé (Itérations 2000-3000).*

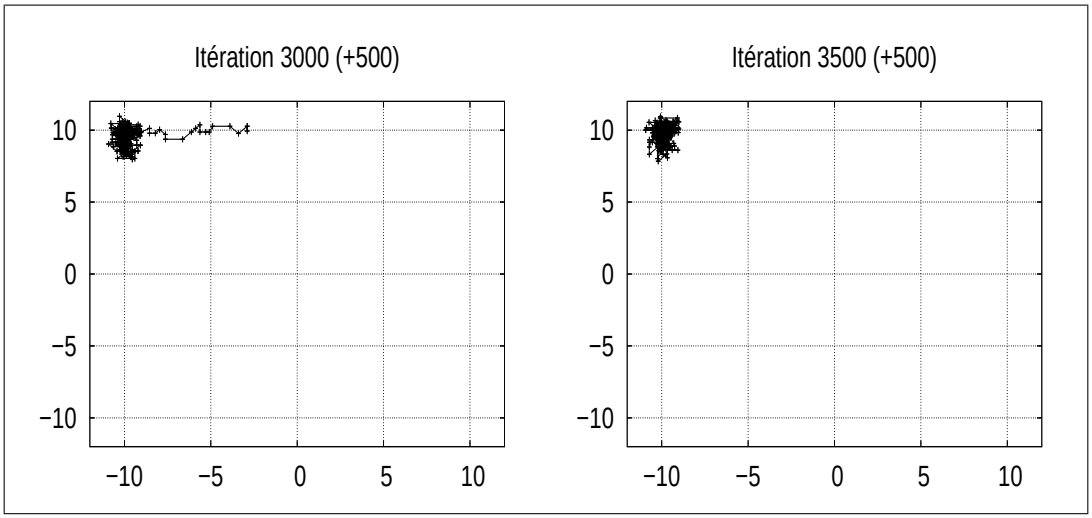


Figure 7.3.4 – *À l'itération 3000 le contrôle reprend le dessus et fait converger l'organisation vers sa première forme, l'objet vers les coin supérieur gauche.(Itérations 3000-3500)*

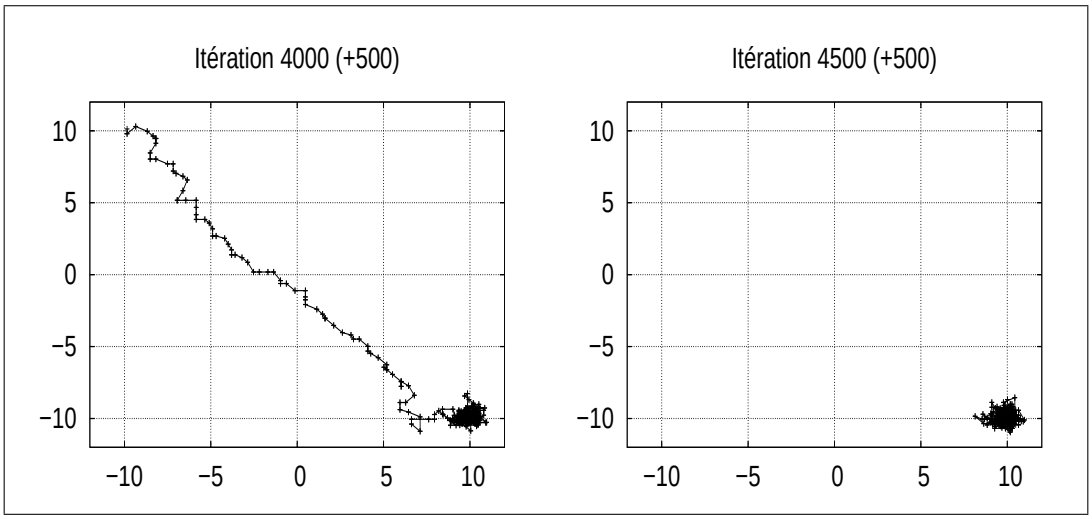


Figure 7.3.5 – *La simulation se termine sur une dernière transition (Itérations 4000-4500)*

La figure 7.3.6 illustre la même trajectoire en décomposant la position de l'objet selon son abscisse et son ordonnée en fonction du temps. Les itérations entre

0 et 1000 correspondent à une phase d'apprentissage. À partir de l'itération 1000, il s'agit de la phase de contrôle, correspondant à l'oscillation forcée de l'organisation, où la trajectoire de l'objet est à à peu près périodique. Nous pouvons constater que le contrôle n'est pas strict. Entre les itérations 1500 et 2000, le système diverge un peu de son état forcé (l'ordonnée diminue un peu). La transition « en bas à droite » vers « en haut à gauche » a lieu un peu avant que le contrôle force cette transition, à l'itération 3000.

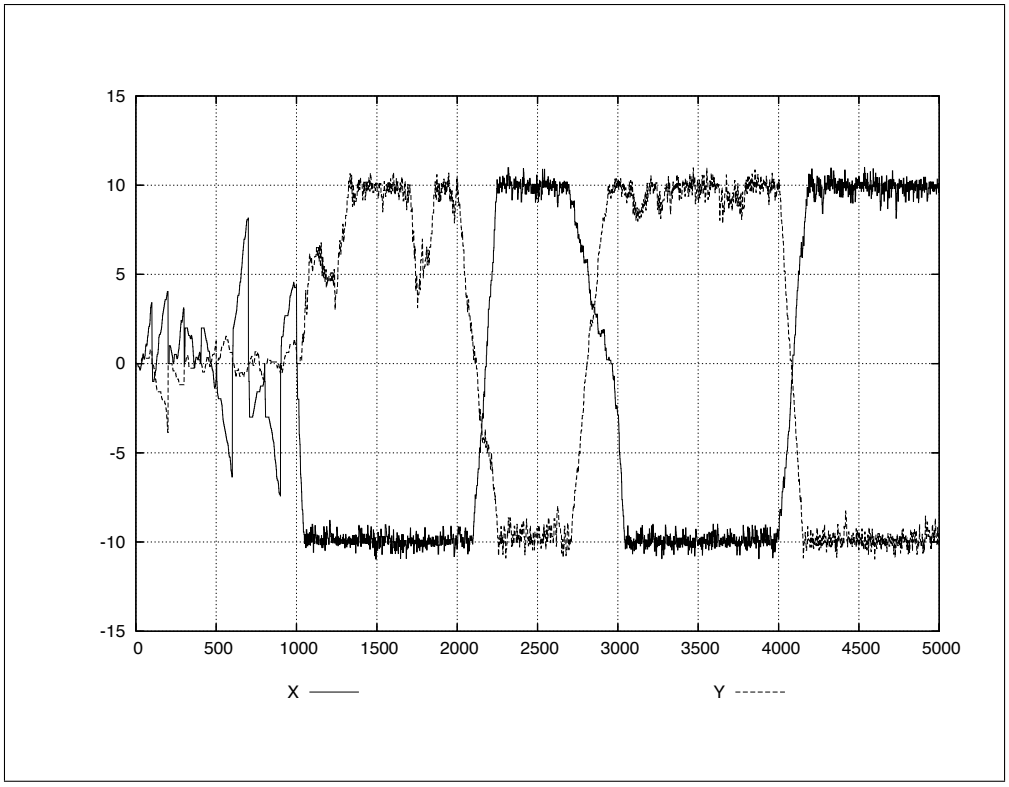


Figure 7.3.6 – *Abscisse et ordonnée de la trajectoire de l'objet en fonction du temps. On peut apercevoir le léger décrochage du système de contrôle entre l'itération 1500 et l'itération 2000 : l'ordonnée Y décroît et croit rapidement.*

Seconde expérience Dans cette seconde expérience, la procédure est à peu près la même mais avec un cycle à quatre formes. Ces formes correspondent au fonctions de déplacement vers le haut à gauche, vers le bas à gauche, vers le bas à droite et vers le haut à droite.

Les figures 7.3.7 à 7.3.14 suivantes illustrent une séquence du pilotage du

système.

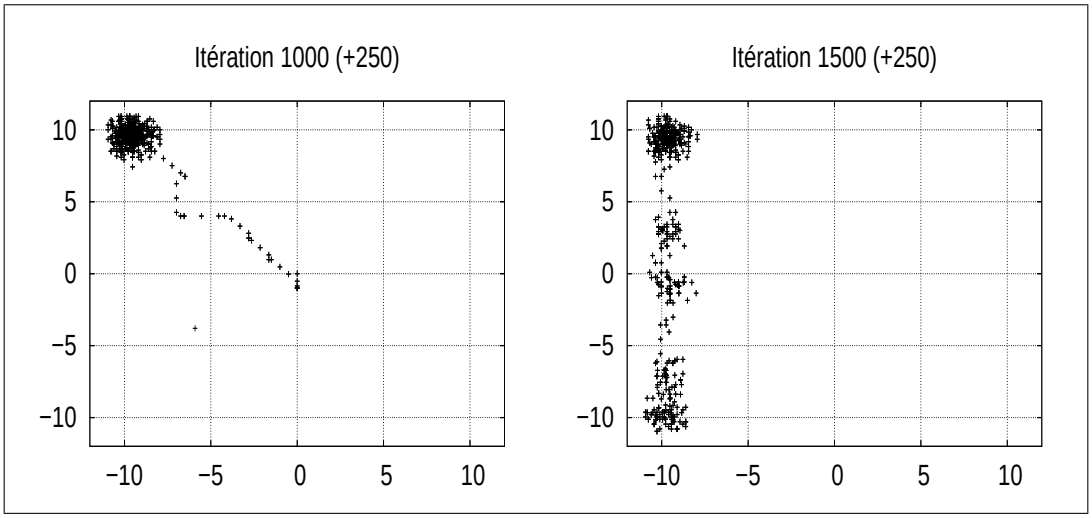


Figure 7.3.7 – *L'expérience débute avec l'objet situé en* $(0,0)$. *Le système, sous contrôle commence à déplacer l'objet vers la première position, en haut à gauche (itérations 1000-1500). Une fois que l'objet a été déplacé dans la première position, le système déplace l'objet vers la seconde position (en bas à gauche, itération 1500-2000.*

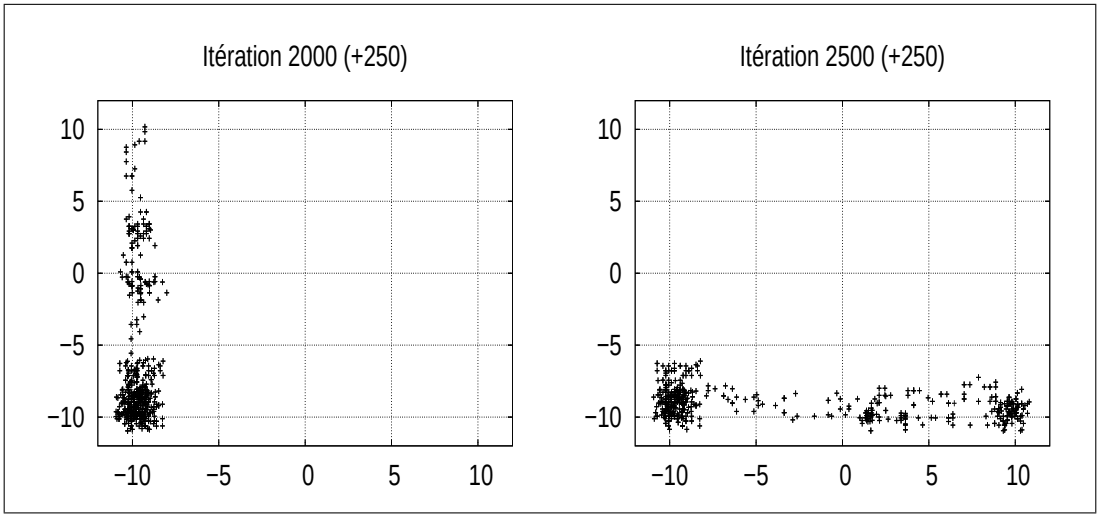


Figure 7.3.8 – *L'objet est ensuite déplacé vers la deuxième position, d'en bas à gauche vers en bas à droite (itérations 2000–3000).*

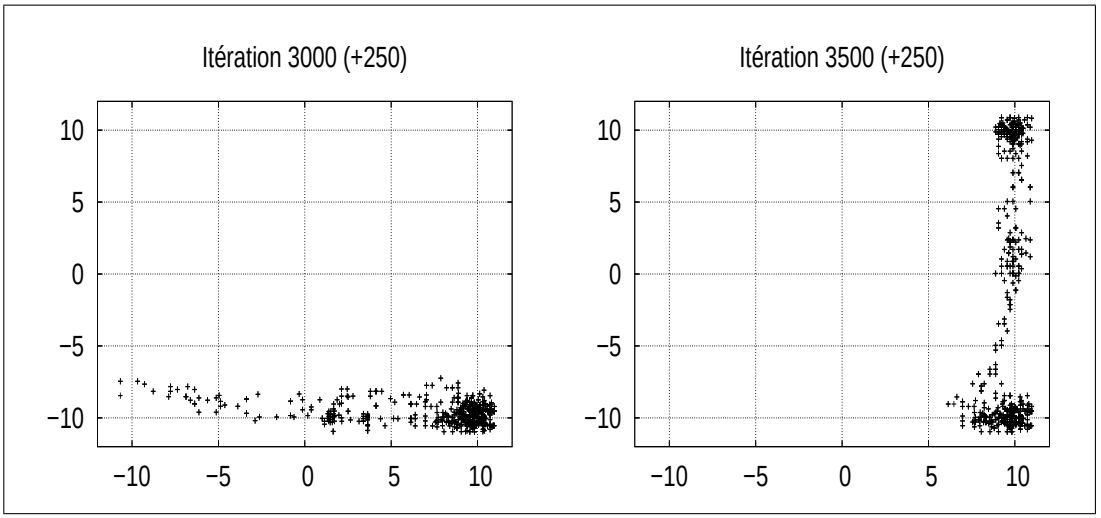


Figure 7.3.9 – *L'objet est ensuite déplacé vers la quatrième position, en haut à droite (itérations 3000-4000).*

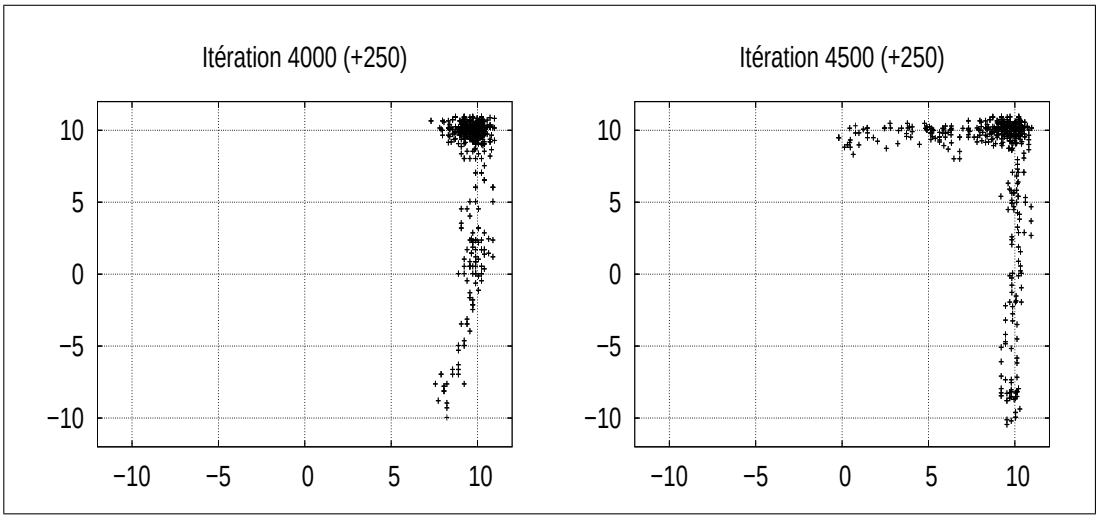


Figure 7.3.10 – *Une fois l'objet est passé par la quatrième position, le système tente de dé placé l'objet vers la première position.*

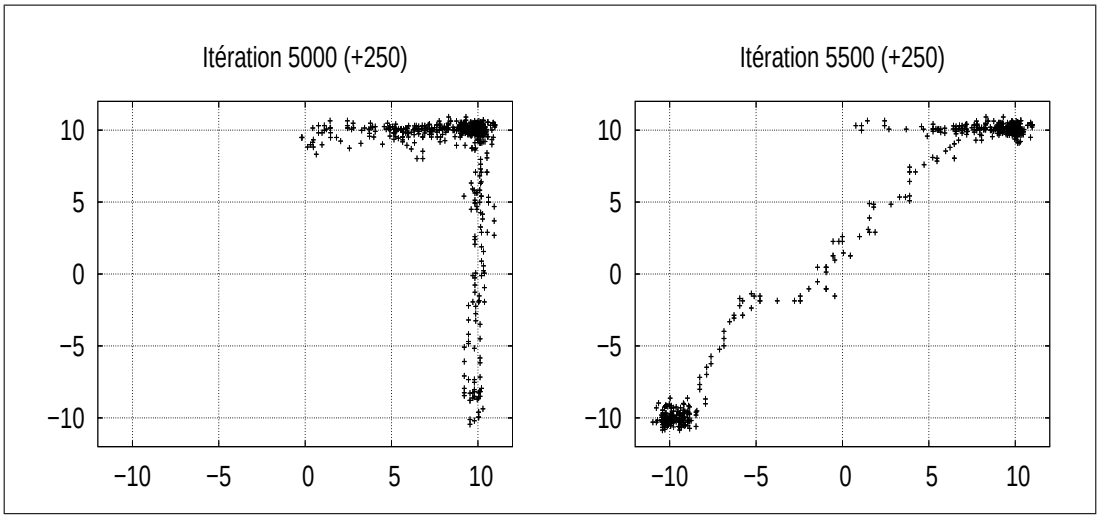


Figure 7.3.11 – *Le contrôle morphologique tente toujours de faire en sorte que le système multi-agent déplace l'objet vers la gauche, mais le contrôle est, ici, mis en défaut. Le système passe à la prochaine position, en bas à gauche.*

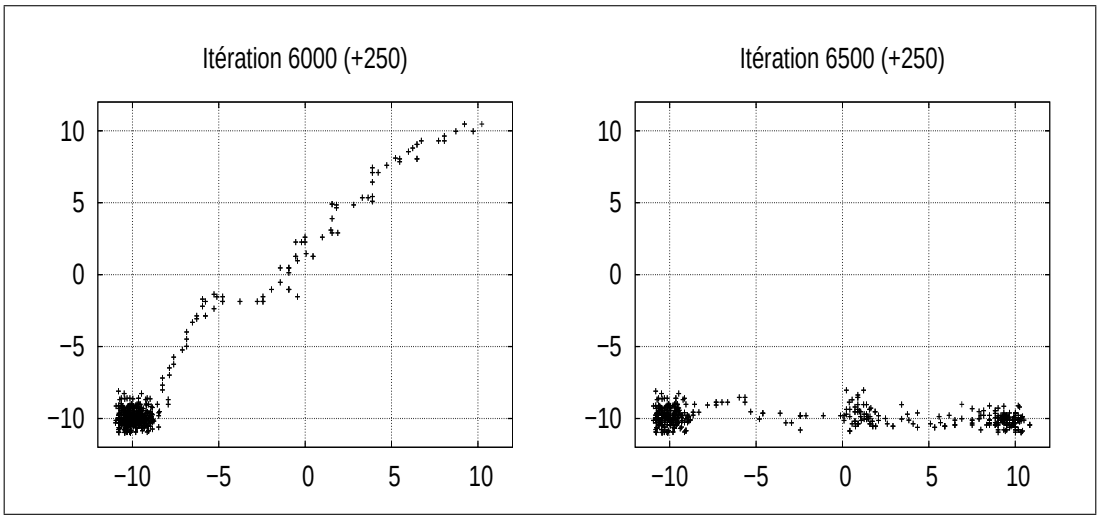


Figure 7.3.12 – *L'objet est finalement déplacé directement vers la deuxième position (en bas à gauche). Puis la boucle continue, l'objet est ensuite déplacé vers la droite...*

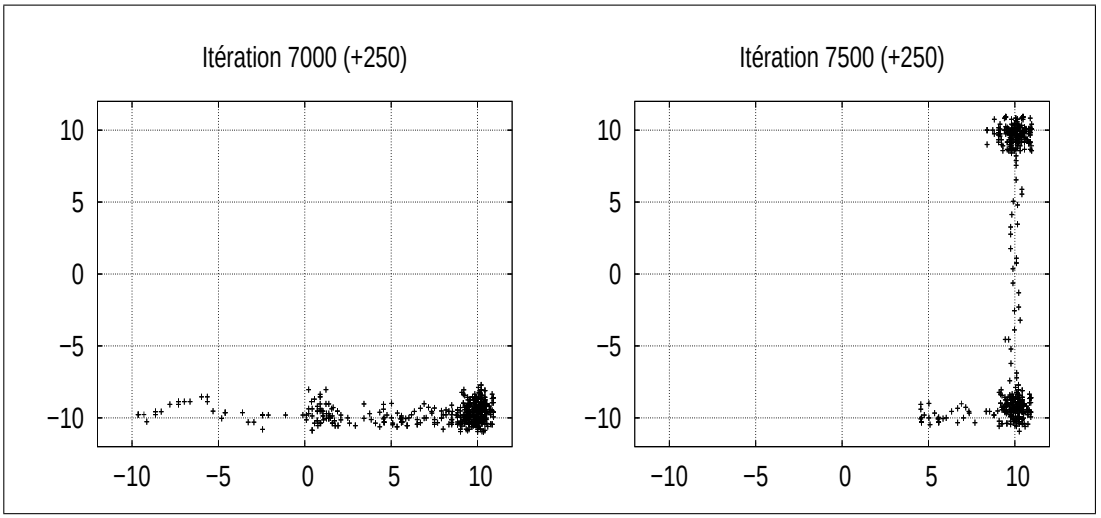


Figure 7.3.13 – *... puis vers le haut...*

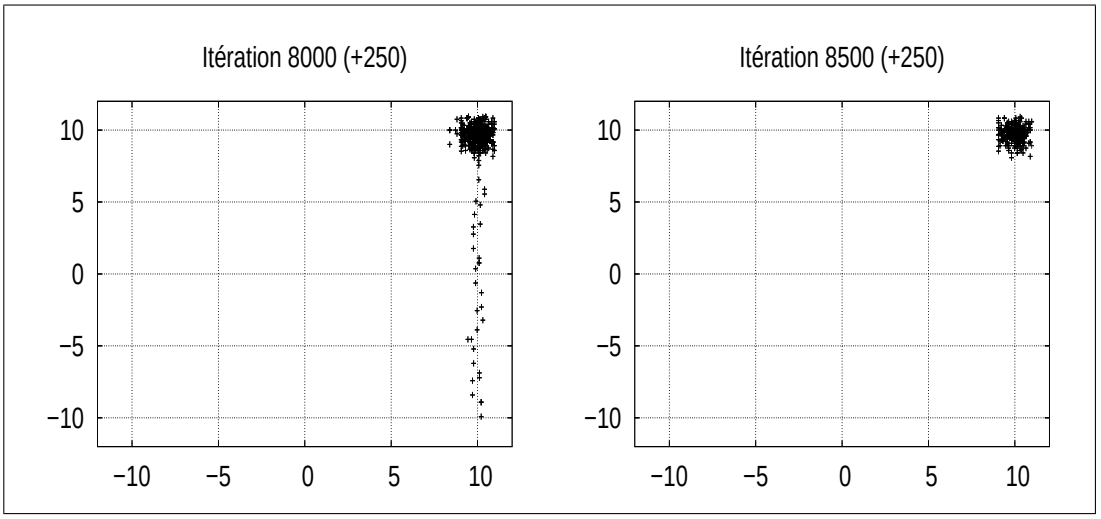


Figure 7.3.14 – *... et finalement l'objet arrive à nouveau à la dernière position.*

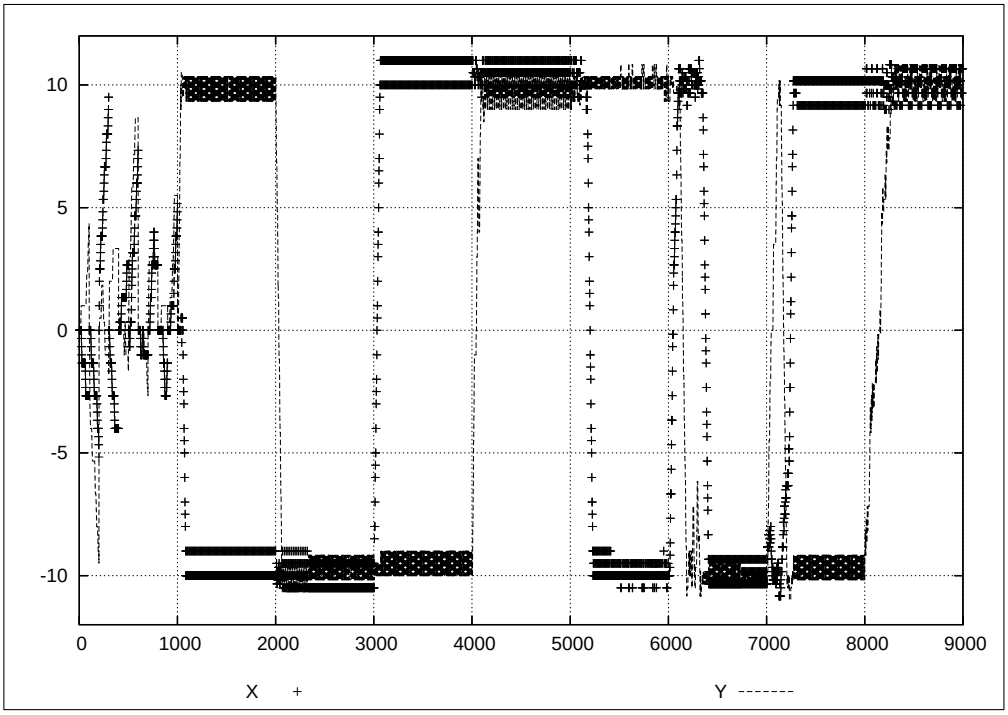


Figure 7.3.15 – *Abscisse et ordonnée de la trajectoire de l'objet en fonction du temps.*

Remarques Ces deux exemples illustrent que la notion d'oscillateur calculable dans deux cas de figures simples. Il est vrai que le système est quelque peu linéaire. Néanmoins cette notion est généralisable à toute ontologie (rôles des agents). Dans notre cas, nous avons associer aux agents les rôles de déplacement d'objets, mais nous pouvons en associer d'autres. Le but de ces exemples est d'illustrer la faisabilité de cette approche de contrôle.

Plans et comportements basés sur l'émotion

Nous allons considérer un système qui planifie l'action comportementale en générant une succession de plans partiels (locaux fonctionnellement) non indépendants et engageant à un ensemble d'actes adaptés à la situation courante, mais avec un objectif global (ou plusieurs cohérents entre eux). Nous posons l'hypothèse suivante, sur la génération des émotions :

> Toute émotion est l'émergence hégémonique d'un cycle global d'oscillations dont la forme décomposable reste typique, et qui altère significativement toute réaction automatique ou strictement rationnelle du système pendant son activité.

Avec cette position, une émotion est une altération d'un comportement « automatique », réactif, objectif du robot placé dans son environnement. Cette altération amplifie certains caractères et en inhibe certains autres, permettant le développement d'un type de comportement adapté à des entrées. On pourra donc se baser sur l'architecture suivante du système :

1. un substrat fonctionnel, qui peut être soit logiciel (comme une organisation aspectuelle) soit matériel, ou bien les deux ;
2. un niveau de génération de plans locaux, partiels, de toutes les échelles, plans prévus pour se coordonner, réaliser un ordonnancement dynamique et s'activer dans des gammes de situations bien prévues. Toute activité du système sera donc un certain ordonnancement de tâches fonctionnelles que l'on appellera un plan d'action ;
3. un niveau émotionnel qui se situe entre les deux niveaux précédents et toute sortie, et qui altère la production de plans réactifs pour produire des sorties adaptées à la situation courante.

Dans un système classique, le niveau émotionnel est réduit à une couche méta qui opère des décisions adéquates dans l'ordonnancement de la planification dynamique. Dans notre système, la génération de l'ensemble des plans courants est modifiée : les plans sont réécrits à chaque fois et ordonnancés de manière adéquate. L'architecture du système est donc génératrice, essentiellement plastique et non simplement structurelle.

Conclusion

Conclusion

Dans cette thèse, nous nous sommes intéressés au problème de l'observation, de l'analyse et du contrôle de systèmes multi-agents.

Nous avons constaté que ces aspects des systèmes multi-agents étaient souvent pas pris en compte par la majeure partie des méthodes de conception et de construction de tels systèmes. Les méthodes orientées génie logiciel estiment cette question réglée à priori par une analyse fine du système. Les méthodes formelles s'intéressent à l'analyse (dans le sens vérification) mais elles se confrontent à la complexité des calculs à effectuer.

Le modèle proposé d'analyse (observation) et de pilotage de systèmes multi-agents est fondé sur la notion de morphologie. L'analyse morphologique consiste en une description géométrique de l'organisation des agents. Notre approche s'apparente aux méthodes employées pour analyser les systèmes complexes dans d'autres domaines de la Science, notamment en Science Physique, où des méthodes de contrôle de système complexes, exploitant la morphologie sont proposées.

Nous avons présenté deux réalisations de contrôle morphologique, une portant sur un système plutôt réactif, et la seconde sur un système comportant des agent un peu plus complexes (sans pour autant être des agents cognitifs). Ces réalisations, qui sont des prototypes, ont montré la faisabilité d'un tel contrôle morphologique. Afin d'espérer un jour pouvoir étayer une théorie générale du contrôle des systèmes multi-agents fondé sur la morphologie, il sera nécessaire de mener davantage de recherches afin d'approfondir certaines notions.

En perspectives, nous avançons que le paradigme des systèmes multi-agents offre un cadre intéressant à l'étude des émotions artificielles, par la voie constructiviste et non réductrice. Les travaux de cette thèse s'inscrivaient initialement dans

un projet à long terme portant sur la réalisation d'un système de contrôle d'un robot sous le paradigme système multi-agent et en exploitant la notion d'émotion artificielle, qui sera finalement mené dans le cadre d'un projet industriel en réalisant un prototype sur un robot « Aibo », le « robot chien » de Sony.

Annexes

Publications

1. Alain Cardon, Jean-Charles Campagne, Mickael Camus, *A self-adapting system generating intentional behavior and emotions*, WRAC 2005 : Workshop on Radical Agent Concept, 2005.
2. Alain Cardon et Jean-Charles Campagne, *A self-adapting system generating emotion*, SID 2005 : Social intelligence design workshop (SID 2005), 2005.
3. Jean-Charles Campagne, Alain Cardon, Toyoaki Nishida, et Étienne Collomb, *Using morphology to control a multi-agent system, an example*, 2nd European Starting AI Researcher Symposium (STAIRS'04), 2004.
4. Jean-Charles Campagne, Alain Cardon, Etienne Collomb, *Analyse et pilotage de systèmes multi-agents massifs*, JFSMA 2004, Journées Francophones sur les Systèmes Mutli-Agents, 2004.
5. Jean-Charles Campagne, Alain Cardon, Toyoaki Nishida, et Étienne Collomb, *Using morphology to control multi-agent systems*, Third NASA-Goddard / IEEE workshop on formal approaches to agent-based systems "FAABS III", 2004.
6. Jean-Charles Campagne, Alain Cardon, Toyoaki Nishida, et Étienne Collomb, *Using morphology to analyse et steer large multi-agent systems at runtime*, SELMAS 2004.
7. Jean-Charles Campagne et Alain Cardon, *An approach to control multi-agent systems*, International IPSI-2003k Conference, 2003.
8. Jean-Charles Campagne et Alain Cardon, *Artificial emotions for robots using massive multi-agent systems*, Social Intelligence Design International Conference, 2003.
9. Alain Cardon et Jean-Charles Campagne, *Artificial sensations for social autonomous robots*, Social Intelligence Design International Conference, 2003.

Table des figures

1.2.1 SMA : Méthode Gaia 8
1.2.2 SMA : Méthode MaSE 10
1.2.3 SMA : Méthode MESSAGE/UML. 12
1.2.4 SMA : Méthode MESSAGE 12
1.2.5 SMA : Méthode Tropos 14
1.2.6 SMA : Méthode Prometheus 16
1.2.7 SMA : Méthode Voyelles 18
1.2.8 SMA : modèle AGR 19
1.2.9 MadKit : structure générale 21
1.2.10 Madkit 22
1.2.11 SMA : Méthode ADELFE 24
1.2.12 SMA : Méthode ZEUS : agent générique 26
1.2.13 SMA : Méthode ZEUS : architecture d'un agent 26
1.2.14 SMA : Terán : trajectoire de simulation 29
1.2.15 SMA : Terán, enveloppe 30
2.2.1 Morphologie de crânes 37
2.2.2 Proies-prédateurs : variation temporelle 40
2.2.3 Proie-prédateur : cycle 41
2.2.4 Proies-prédateurs : tendance du système 41
2.2.5 Proies-prédateurs : friction 42
2.2.6 Proies-prédateurs : modèle plus adéquat 42
2.2.7 Pendule, espace des phases 43
2.2.8 Diagramme de l'eau 45
2.2.9 Fonction logistique 48
2.2.10 Attracteurs de GNPA 53
2.2.11 Méthode du retard 54
2.2.12 Lesage : architecture système 57
2.2.13 Lesage : notion d'importance 59
2.2.14 Lesage : séquence paysage 60
3.1.1 Espace morphologique 68

3.1.2 Espace morphologique 68
3.1.3 Espace morphologique 69
3.1.4 Espace morphologique 69
3.1.5 Espace morphologique, trajectoire spontanée 71
3.1.6 Modèle : niveaux de système 72
3.1.7 Espace morphologique 73
3.1.8 Espace morphologique 74
3.2.1 Modèle : corrélation entre forme et fonction 75
3.2.2 L'idée sous-jacente du modèle 76
3.3.1 Couplage organisation aspectuelle-environnement 77
3.4.1 Projection 80
3.4.2 Mesure et paramétrage 81
3.5.1 Graphe d'évolution 84
3.5.2 Graphe d'évolution 85
3.5.3 Graphe d'évolution 88
3.5.4 Graphe d'évolution, comportement 89
3.6.1 Composant 90
3.6.2 Contrôle 92
4.2.1 Exemple réactif : interactions agents 97
4.2.2 Exemple réactif : niveau d'énergie 98
4.2.3 Exemple réactif : organigramme agent 102
4.3.1 Exemple réactif : pré-traitement de la forme 104
4.3.2 Exemple réactif : transformation d'histogramme 106
4.3.3 Exemple réactif : règles élémentaires 107
4.3.4 Exemple réactif : classification fine 108
4.3.5 Exemple réactif : convergence 109
4.3.6 Exemple réactif : divergence 110
4.4.1 Exemple réactif : contre hypothèse 111
4.4.2 Exemple réactif : exemple de courbes 112
4.4.3 Exemple réactif : Distribution de l'écart 114
4.4.4 Exemple réactif : $\frac{v_m}{v}$, binaires, densité 116
4.4.5 Exemple réactif : $\frac{v_m}{v}$, ternaires, densité 117
4.4.6 Exemple réactif : $\frac{v_m}{v}$, binaires, marges 119
4.4.7 Exemple réactif : $\frac{v_m}{v}$, ternaires, marges 120
4.4.8 Exemple réactif : $\frac{v_m}{v}$, binaires, cibles 122
4.4.9 Exemple réactif : $\frac{v_m}{v}$, ternaires, cibles 123
4.4.10 Exemple réactif : quelques formes. 125
4.4.11 Exemple réactif : écart en fonction des formes 125
5.2.1 Prototype : Schéma d'un composant 130
5.2.2 Prototype : adresses 132
5.2.3 Prototype : structure message 133

5.2.4 Prototype : objets dans environnement 136
5.2.5 Exemple ontologie 138
5.2.6 Exemple espace objets 141
5.2.7 Agent aspectuel : rôle 142
5.2.8 Agent aspectuel : rôle et agent 142
5.2.9 Agent aspectuel : agent groupe 144
5.3.1 Agent aspectuel : organigramme du comportement 146
5.3.2 Agent aspectuel : état de veille 147
5.3.3 Préférence asymétrique 150
5.3.4 Agent aspectuel : Groupe 153
5.3.5 Espace objets 154
5.3.6 Intégration groupe 155
5.3.7 Intégration réussie 156
5.3.8 Intégration échouée 157
5.3.9 Groupe sortie volontaire 158
5.3.10 Groupe éjection 158
5.3.11 Fusion de groupes 159
5.3.12 Vie de groupe 160
5.4.1 Capture écran dévermineur 161
5.4.2 Capture écran dévermineur 162
5.6.1 Construction du graphe 166
5.6.2 Approximation du graphe 167
5.6.3 Comparaison de graphes 168
5.6.4 Boucle de contrôle 169
6.1.1 Schéma de l'application 172
6.1.2 Ontologie 173
6.2.1 Propositions de mouvement individuel 175
6.2.2 Propositions de mouvement avec groupe 177
6.2.3 Morphologie structure fonction 178
6.3.1 Exemple : objet dans l'environnement 179
6.3.2 Exemple de mouvements 180
6.3.3 Exemple de mouvements 181
6.3.4 Exemple de mouvements 182
6.3.5 Traces 183
6.3.6 Exemple de graphes d'évolution 184
6.3.7 Traces 185
6.3.8 Évolution nombre formes 186
6.3.9 Distribution des formes 187
6.3.10 Distribution $\frac{f_1}{f_2}$ 188
6.3.11 Abscisse moyenne sans contrôle 188
6.3.12 Coordonnée moyenne sans contrôle 189

6.3.13 Abscisse moyenne avec contrôle 190
6.3.14 Coordonnée moyenne avec contrôle 191
6.3.15 Comparaison 191
6.3.16 Comparaison 192
6.3.17 Comparaison 192
6.3.18 Comparaison 193
6.3.19 Comparaison 193
6.3.20 Distribution des formes 194
6.3.21 Graphe d'évolution. 195
6.3.22 Graphe d'évolution. 196
6.3.23 Évolution de l'abscisse. 196
7.1.1 Émotion théorie : James-Lange 203
7.1.2 Émotion théorie : Cannon-Bard 204
7.1.3 Émotion théorie : Schachter-Singer 204
7.1.4 Émotion théorie : Lazarus 205
7.1.5 Coupe cerveau 206
7.1.6 Émotions : mécanisme 209
7.2.1 Émotion modèle : Ventura 214
7.2.2 Émotion modèle : Velásquez 215
7.2.3 Modèle émotions artificielles : Gadanho 219
7.2.4 Émotions artificielles : Gadanho 220
7.2.5 H-CogAff 222
7.3.1 Oscillateur calculable 227
7.3.2 Séquence 1 230
7.3.3 Séquence 1 230
7.3.4 Séquence 1 231
7.3.5 Séquence 1 231
7.3.6 Oscillations 232
7.3.7 Séquence 2 233
7.3.8 Séquence 2 233
7.3.9 Séquence 2 234
7.3.10 Séquence 2 234
7.3.11 Séquence 2 235
7.3.12 Séquence 2 235
7.3.13 Séquence 2 236
7.3.14 Séquence 2 236
7.3.15 Oscillations 237
7.5.1 Boucle de pilotage 264
7.7.1 Photo d'un Aibo 265
7.8.1 Membres de l'Aibo. 268
7.9.1 Boucle de pilotage, organisations d'agents. 269

7.9.2 Rôle 270
7.10.1 Capture d'écran 273
7.10.2 Capture d'écran 274
7.10.3 Capture d'écran 275
7.10.4 Capture d'écran 276
7.10.5 Capture d'écran 277

Bibliographie

[ade] *Adelfe*, http://www.irit.fr/ADELFE/. 1.2.1

[Age] AgentSheets Inc., *Agentsheets*, http://agentsheets.com/. 1.2.1

[AI] AI center, *Excalibur agent*, http://www.ai-center.com/projects/excalibur/publications.html. 1.2.1

[And03] Michael L. Anderson, *Embodied cognition : a field guide*, Artificial Intelligence **149** (2003), no. 1, 91–130. 7.2.1

[AS92] Ralph H. Abraham and Christopher D. Shaw, *Dynamics : the geometry of behavior*, 2 ed., Addison-Wesley publishing company, 1992. (document)

[Ash62] Ross Ashby, *Principles of self-organizing systems*, Heinz Von Foerster and George W. Zopf, Jr. (eds.), Principles of self-organization, 1962. 3.1.2

[AT00] Nader Azarmi and Simon Thompson, *Zeus : A toolkit for building multi-agent systems*, Proceedings of fifth annual Embracing Complexity conference, Paris, 2000. 1.2.1

[Auv05] Jean-Marie Auvinet, *Interprétation, intéractions et connaissance – artefacts et assisatance aux opérateurs radio des salles d'informations et de commandement*, Ph.D. thesis, Université Paris I - Sorbonne, 2005. 2.2.4

[Bat97] J. Bates, *The role of emotion in believable agents*, Communications of the ACM **37** (1997), no. 7, 122–125. 7.2.1

[BD01] Jean-Pierre Briot and Yves Demazeau, *Principes et architectures des systémes multi-agents*, Hermés-Lavoisier, 2001. 1.1.2

[Ben02] Endika Bengoetxea, *Mise en correspondence inexacte de graphes par algorithmes d'estimation de distributions*, Ph.D. thesis, École Supérieure Nationale des Télécommunications, 2002. 3.4.2, 5.6.2

[Bir] University of Birbingham, *The simagent toolkit (sim_agent)*, http://www.cs.bham.ac.uk/~axs/cog_affect/sim_agent.html. 1.2.1, 7.2.2

[BK99] D. S. Boukal and V. Krivan, *Lyapunov functions for lotka-volterra predator-prey models with optimal foraging behavior*, J. Math. Biol. **39** (1999), no. 6, 493–517. 2.2.2

[Bol] DEIS Universitad di Bologna, *Tucson*, http://www-lia.deis.unibo.it/Research/TuCSoN/. 1.2.1

[Boo91] F. L. Bookstein, *Morphometric tools for landmark data : geometry and biology*, Cambridge Univ. Press : New York, 1991. (document), 2.2.2

[BPST02] M. Benerecetti, M. Panti, L. Spalazzi, and S. Tacconi, *Verification of payment protocols via multiagent model checking*, CAISE 2002, LNCS 2248 (A. Banks Pidduck et al., ed.), 2002, pp. 311–327. 1.2.2

[Bry86] R. E. Bryant, *Graph-based algorithms for boolean function manipulation*, IEEE Transaction on computers (1986), 667–691. 1.2.2

[BTe] BTexact Technologies Executive, *Zeus*, http://www.labs.bt.com/projects/agents/zeus/. 1.2.1, 1.2.1

[Can27] W. B. Cannon, *The james-lange theory of emotion : A critical examination and an alternative theory*, American Journal of Psychology **39 :10-124** (1927). 7.1.2

[Car] Carnegie Mellon University, *Retsina*, http://www-2.cs.cmu.edu/~softagents/retsina_agent_arch.html. 1.2.1

[Car99] Alain Cardon, *Conscience artificielle et systèmes adaptatifs*, Eyrolles, December 1999. (document), 2.2.4, 3.3, 3.4.1, 5, 5.1

[Car03] _____, *Modéliser et concevoir une machine pensante. approche constructible de la conscience artificielle*, Éditions Automates Intelligents, March 2003. (document), 2.2.4, 3.3, 3.4.1, 3.7.1

[Car05] _____, *Le complexité organisée. systèmes adaptatifs et champ organisationnel*, Lavoisier, 2005. 1.1.1

[CC05] Alain Cardon and Jean-Charles Campagne, *A self-adapting system generating emotion*, SID 2005 : Social intelligence design workshop, 2005. 7.3.3

[CCG+01] Giovanni Caire, Wim Coulier, Francisco J. Garijo, Jorge Gomez, Juan Pavon, Francisco Leal, Paulo Chainho, Paul E. Kearney, Jamie Stark, Richard Evans, and Philippe Massonet, *Agent oriented*

analysis using message/UML, AOSE, 2001, pp. 119–135. (document), 1.2.1, 1.2.1

[CH98] Andrew D. J. Cross and Edwin R. Hancock, *Graph matching with a dual-step em algorithm*, IEEE transactions on pattern analysis and machine intelligence **20** (1998), 1236–1253. 3.4.2, 5.6.2

[Chi] University of Chicago, *Repast*, `http://repast.sourceforge.net/`. 1.2.1

[Cin] University of Cincinnati, *Jafmas / jive*, `http://www.ececs.uc.edu/~abaker/JiVE/`. 1.2.1

[CIR] CIRAD, *Cormas*, `http://cormas.cirad.fr/`. 1.2.1

[CKP95] W. J. Christmas, J. Kittler, , and M. Petrou, *Structural matching in computer vision using probabilistic relaxation*, IEEE transactions on pattern analysis and machine intelligence **17** (1995), no. 8. 3.4.2, 5.6.2

[CL90] P. R. Cohen and H. J. Levesque, *Intention is choice with commitment*, Artificial Intelligence (1990), no. 42, 213–261. 1.2.2

[Com] Comet Way Ltd., *Comet way*, `http://www.agentkernel.com/`. 1.2.1

[Dam94] A. Damasio, *Descartes' error.*, New York : Grosset-Putman, 1994. 7.2.1, 7.2.2, 7.2.2

[Dam00] Antonio R. Damasio, *The feeling of what happens : body and emotion in the making of consciousness*, Harcourt, 1999, 2000. 7.1.1, 7.1.3, 7.1.3, 7.1.4, 7.1.5, 7.2.1, 7.3.1

[Dam04] Damasio, *Looking for Spinoza : joy, sorrow, and the feeling brain*, Harcourt Inc., 2004. 7.2.1

[DAR] DARPA, *Courgaar*, `http://www.cougaar.org/`. 1.2.1

[Del] University of Delaware, *Decaf*, `http://www.eecis.udel.edu/~decaf/`. 1.2.1

[Del01] S. Deloach, *Analysis and design using mase and agenttool*, 12th Midwest artificial intelligence and cognitive science conference (MAICS), 2001. (document), 1.2.1, 1.2.1

[Dem01] Yves Demazeau, *« voyelles », rapport d'habilitaion à diriger des recherches*, Tech. report, Institut National Polytechnique de Grenoble, Laboratoire Leibniz, Grenoble, Avril 2001. (document), 1.2.1, 1.2.1

[DJD+un] Riviere D, Mangin JF, Papadopoulos-Orfanos D, Martinez JM, Frouin V, and Regis J, *Automatic recognition of cortical sulci of*

the human brain using a congregation of neural networks, Medical Image Analysis (2002 Jun), 6(2) :77–9. 3.4.2, 5.6.2

[dMV00] Rodrigo Martins de Matos Ventura, *Emotion-based agents*, Master's thesis, Lisbonne, 2000. 7.2.2

[Dur99] Stéphane Durand, *Représenation de points de vues multiples dans une situation d'urgence : modélisation par organisation d'agents*, Ph.D. thesis, Université du Havre, 1999. (document), 2.2.4

[Emo] Emorphia Ltd., *Fipa-os*, `http://www.emorphia.com/research/about.htm`. 1.2.1

[EUR01a] EURESCOM, *MESSAGE*, `http://www.eurescom.de/~public-webspace/P900-series/P907/backgrou.htm`, 2001. 1.2.1

[EUR01b] _____, *MESSAGE : Methodology for engineering systems of software agent*, `http://www.eurescom.de/public/projects/P900-series/p907/default.asp`, 2001. 1.2.1

[Fer95] Jacques Ferber, *Les systèmes multi-agents*, InterEdition, Paris, 1995. (document), 1.1.1, 3.1.1

[FG96] Stan Franklin and Art Graesser, *Is it an agent, or just a program ? : a taxonomy for autonomous agents*, Proceedings of the third international workshop on agent theories, architectures, and languages (Springer-Verlag, ed.), 1996. (document)

[FG99] J. Ferber and O. Gutknecht, *Operational semantics of a role-based agent architecture*, ATAL, 1999. 1.2.1

[Gad03] Sandra Clara Gadanho, *Learning behavior-selection by emotions and cognition in a multi-goal robot task*, Journal of Machine Learning Research (2003), 385–412. 7.2.2

[GBH87] Les Gasser, Carl Braganza, and Nava Herman, *Mace : a flexible testbed for distributed ai research*, Distributed artificial intelligence (Michael N. Huhns, ed.), Pitman publishers, 1987, pp. 119–152. 1.2.1

[GF98a] O. Gutknecht and J. Ferber, *A model for social structures in multi-agent systems*, Tech. report, RR.LIRMM 98040, 1998. 1.2.1

[GF98b] _____, *Un modèle d'analyse, de construction et d'exécution de systèmes multi-agents*, Actes des JFIADSMA'98 (Editions Hermès, ed.), 1998. 1.2.1

[GF00] Olivier Gutknecht and Jacques Ferber, *The MADKIT agent platform architecture*, Agents workshop on infrastructure for multi-agent systems, 2000, pp. 48–55. 1.2.1

[GG00] M.-P. Gleizes and P. Glize, *Des systéme multi-agents pour le courtage adaptatif*, Systémes multi-agents – Méthodologie, technologie et expériences – Actes des huitiémes JFIADSMA (S. Pesty et C. Sayettat-Fau, ed.), Hermés, 2000, pp. 117–132. 1.2.1

[GH] S. Gadanho and J. Hallam, *Exploring the role of emotions in autonomous robot learning*. 7.2.2

[GH01a] _____, *Emotion-triggered learning in autonomous robot control*, 2001. 7.2.2

[GH01b] _____, *Robot learning driven by emotions*, 2001. 7.2.2

[GK02] Les Gasser and Kelvin Kakugawa, *MACE3J : fast flexible distributed simulation of large, large grain multi-agent systems*, AAMAS, 2002. 1.2.1

[GLPG00] M.-P. Gleizes, J. Link-Pezet, and P. Glize, *An adaptive multi-agent tool for electronic commerce*, Second international workshop on knowledge media networking, Gettysburg, USA, 2000. 1.2.1

[GMP02] Fausto Giunchiglia, John Mylopoulos, and Anna Perini, *The tropos software development methodology : processes, models and diagrams*, Proceedings of the first international joint conference on autonomous agents and multiagent systems, ACM Press, 2002, pp. 35–36. (document), 1.2.1, 1.2.1

[Gue96] Zahia Guessoum, *Un environnement opérationnel de conception et de réalisation de systémes multi-agents*, Ph.D. thesis, Laforia, 1996. 1.2.1

[Gut] Johannes Gutenberg-Universit Mainz, *Xraptor*, `http://www.informatik.uni-mainz.de/~polani/XRaptor/XRaptor.html`. 1.2.1

[Gut01] Olivier Gutknecht, *Proposition d'un modèle organisationnel générique de systèmes multi-agents. examen de ses conséquences formelles, implémantatoires et méthodologiques*, Ph.D. thesis, Université Montpellier II, 2001. 1.2.1, 1.2.1

[Hil94] Hilborn, *Chaos and nonlinear dynamics*, New York Oxford, Oxford University Press, 1994. 2.2.3

[HSNL98] D. T. Ndumu H. S. Nwana and L. C. Lee, *ZEUS : an advanced toolkit for engineering distributed multi-agent systems*, 3rd int. conf. practical appl. intelligent agents and multi-agent technology, 1998, pp. 377–391. 1.2.1

[IKV] IKV++ Technologies AG, *Grasshopper*, `http://www.ikv.de/content/Grasshopper/Grasshopper_e.htm`. 1.2.1

[Jam90] William James, *The principles of psychology*, 1890. 7.1.2

[jat] *Jatlitebean*, http://kmi.open.ac.uk/people/emanuela/JATLiteBean/. 1.2.1

[JP03] Georgé Jean-Pierre, *The amas theory for complex problem solving based on self-organizing cooperative agents*, First european workshop on multi-agent systems (Oxford, UK), vol. 1, 2003. 1.2.1

[JPMPPC03] Georgé Jean-Pierre, Gleizes Marie-Pierre, Glize Pierre, and Régis Christine, *Real-time simulation for flood forecast : an adaptive multi-agent system staff*, Proceedings of the AISB'03 symposium on adaptive agents and multi-agent systems (University of Wales, Aberystwyth), 2003. 1.2.1

[JW98] Nicholas R. Jennings and Michael J. Wooldridge, *Applications of intelligent agents*, Agent Technology Foundations, Applications, and Markets , Springer-Verlag, 1998. 1.1.1

[Kaa] *Kaariboga*, http://www.projectory.de/kaariboga/. 1.2.1

[Kana] Kansas State University, *Agenttool*, http://www.cis.ksu.edu/~sdeloach/ai/. 1.2.1

[Kanb] Kansas State university, *Mase*, http://www.cis.ksu.edu/~sdeloach/ai/projects/mase.htm. 1.2.1

[Lab] Sandia National Laboratories, *Jess*, http://herzberg.ca.sandia.gov/jess/. 1.2.1

[Laz91] R. S. Lazarus, *Emotion and adaptation*, New York : Oxford University Press., 1991. 7.1.2

[Les00] Franck Lesage, *Interprétation adaptative du discours dans une situation multiparticipants : modélisation par agents*, Ph.D. thesis, Université du Havre, 2000. (document), 2.2.4, 2.2.13, 2.2.14, 3.7.1

[Lew92] F.L. Lewis, *Applied optimal control and estimation*, Prentice-Hall, 1992. (document)

[Loga] Logica UK Ltd, *Cable*, http://public.logicacmg.com/~grace/Architecture/Cable/public/. 1.2.1

[Logb] Logilab, *Narval*, http://www.logilab.org/projects/narval/. 1.2.1

[Ltda] Agent-Lab Ltd., *Maml*, http://www.maml.hu/. 1.2.1

[Ltdb] Agent Oriented Software Pty. Ltd, *Jack*, http://www.agent-software.com/shared/home/. 1.2.1

[Mac] *Mace / mace3j*, http://www.isrl.uiuc.edu/amag/mace/. 1.2.1

[Mada] *Madkit*, http://www.madkit.org/. 1.2.1, 1.2.1

[Madb] Universitad de Madrid, *Mast*, http://www.gsigma-grucon.ufsc.br/massyve/mkit.htm. 1.2.1

[Mar] University of Maryland, *Impact*, http://www.cs.umd.edu/projects/impact/. 1.2.1

[MB00] Bruno T. Messmer and Horst Bunke, *Efficient subgraph isomorphism detection : A decomposition approach*, IEEE Transactions on Knowledge and Data Engineering **12** (2000), 307 – 323. 3.4.2, 5.6.2

[MH87] Fionn Murtagh and A. Heck, *Multivariate data analysis*, Kluwer Academic, Dordrecht, 1987. (document), 3.4.2, 3.5.3

[Min03] Minsky, *The emotion machine*, Version préliminaire du livre en ligne : http://web.media.mit.edu/~minsky/, 2003. 7.2.1, 7.2.2

[MIT] MIT MEDIA lab, *Starlogo*, http://education.mit.edu/starlogo/. 1.2.1

[Mot] Motorola and ADAC and Broadcom and BT and TILabs Uni Parma and Siemens, *Jade / leap*, http://leap.crm-paris.com/, http://sharon.cselt.it/projects/jade/. 1.2.1

[NCN98] D. T. Ndumu, J.C. Collis, and H. S. Nwana, *Towards desktop personal travel agents*, BT Technology Journal **16** (1998), no. 3, 69–78. 1.2.1

[nus] *Nusmv*, http://nusmv.irst.itc.it/. 1.2.2

[Nwa98] Hyacinth S. Nwana, *Agent-mediated electronic commerce : issues, challenges and some viewpoints*, Proceedings of Agents '98, Minneapolis, 1998,, pp. 189–196. 1.2.1

[oca] *Ocaml : http://caml.inria.fr*. 4.4

[OE02] Teràn Oswaldo and Bruce Edmonds, *Computational complexity of a constraint model-based proof of the envelope of tendencies in a mas-based simulation model*, Second international workshop on complexity deduction (CiAD) in conjunction with CADE-18 and FloC 2002 (Copenhagen, Denmark), 25-26 July 2002. 1.2.2

[OGY90] E. Ott, C. Grebogi, and J. A. Yorke, *Controlling chaotic dynamical systems*, Controlling Chaos, Phys. Rev. Lett. (1990). 2.2.3

[Pic04] Gauthier Picard, *Méthodologie de développement de systémes multi-agents adaptatifs et conception de logiciels à fonctionnalité émergente*, Ph.D. thesis, Universitée Paul Sabatier de Toulouse III, 2004. 1.2.1

[PSZ99] M. Pelillo, K. Siddiqi, , and S. Zucker, *Continuous-based heuristics for graph and tree isomorphisms, with application to computer vision*, P. M. Pardalos, editor, Approximation and complexity in Numerical Optimization : Continuous and Discrete Problems. Kluwer Academic Publishers (1999). 3.4.2, 5.6.2

[PW02] L. Padgham and M. Winikoff, *Prometheus : a methodology for developing intelligent agents*, Third international workshop on agent oriented software engineering, at AAMAS (Bologna, Italy), July 2002. (document), 1.2.1, 1.2.1

[RD01] P-M. Ricordel and Y. Demazeau, *Madel : un langage descriptif pour le développement et le déploiement de systèmes multi-agents*, Atelier méthodologie et environnements pour les systèmes multi-agents (Grenoble), June 2001. 1.2.1

[RD02a] _____, *La plate-forme volcano : modularité et réutilisabilité pour les systèmes multi-agents*, Numéro spécial sur les plates-formes de développement SMA. Revue Technique et Science Informatiques (TSI), Hermès, Paris, France (2002). 1.2.1

[RD02b] _____, *Volcano : a vowels-oriented multi-agent platform.*, In Proceedings of the International Conference of Central Eastern Europe on Multi-Agent Systems (CEEMAS'01), From Theory to Practice in Multi-Agent Systems, Dunin-Keplicz and Nawarecki, eds, LNAI 2296 (Krakow, Poland) (Springer Verlag, ed.), 2002, pp. 252–262. 1.2.1

[Rec] Recursion Software, Inc., *Voyager*, http://www.recursionsw.com/voyager.htm. 1.2.1

[RG91] Anand S. Rao and Michael P. Georgeff, *Modeling rational agents within a BDI-architecture*, Proceedings of the 2nd international conference on principles of knowledge representation and reasoning (KR'91) (James Allen, Richard Fikes, and Erik Sandewall, eds.), Morgan Kaufmann publishers Inc. : San Mateo, CA, USA, 1991, pp. 473–484. 1.2.2

[Ric01] Pierre-Michel Ricordel, *Programmation orientée multi-agents : développement et déploiement de systémes multi-agents voyelles*, Ph.D. thesis, Institut national polytechnique de Grenoble, Octobre 2001. 1.2.1

[RL04] F. Raimondi and A. Lomuscio, *Symbolic model checking of multi-agent systems using obdds*, Proceedings of the third nasa workshop on formal approaches to agent-based systems, FAABS III (Springer Verlag LNCS, ed.), april 2004. (document), 1.2.2

[RMIa] RMIT University, Melbourne, AUSTRALIA, *Prometheus*, http://www.cs.rmit.edu.au/agents/SAC2/methodology.html. 1.2.1

[RMIb] _____, *Prometheus*, http://www.cs.rmit.edu.au/agents/prometheus/. 1.2.1

[San] Federal University of Santa Catarina, *Massyve*, http://www.gsigma-grucon.ufsc.br/massyve/mkit.htm. 1.2.1

[SCC97] M. Singh, A. Chatterjee, and S. Chaudhury, *Matching structural shape descriptions using genetic algorithms*, Pattern Recognition **30** (1997), no. 9, 1451–1462. 3.4.2, 5.6.2

[Sch99] Heinz G. Schuster (ed.), *Handbook of chaos control*, Wiley-VCH, 1999. (document), 2.2.3, 2.2.3

[Slo01a] A. Sloman, *Varieties of affect and the cogaff architecture schema*, Symposium on Emotion, Cognition, and Affective Computing at the AISB ?01 Convention, march 2001. 7.2.2

[Slo01b] Aaron Sloman, *Beyond shallow models of emotion*, 2001. 7.2.2

[Slo04] _____, *What are emotion theories about ?*, American Association for Artificial Intelligence, 2004. 7.2.5

[Ste] W. Richard Stevens, *Tcp/ip illustrated. volume 1. the protocols*. 2.2.4

[Ter01] Oswaldo Teràn, *Emergent tendencies in multi-agent-based simulations using constraint-based methods to effect practical proofs over finite subsets of simulation outcomes.*, Ph.D. thesis, Manchester Metropolitan University, Manchester, UK, 2001. (document), 1.2.2

[TEW] Oswaldo Terán, Bruce Edmonds, and Steve Wallis, *Determining the envelope of emergent agent behaviour via architectural transformation*. 1.2.2, 1.2.14, 1.2.15

[TFG+99] Xavier Topin, Vincent Fourcassie, Marie-Pierre Gleizes, Guy Theraulaz, Christine Regis, and Pierre Glize, *Theories and experiments on emergent behaviourl : from natural to artificial systems and back*, Proceedings on European Conference on Cognitive Science, Siena, 1999. 1.2.1

[Tho17] D'Arcy Thompson, *On growth and form*, Cambridge University Press, 1917. 2.2.2, 2.2.1

[Tho72] René Thom, *Stabilité structurelle et morphogénèse*, InterEditions, Paris, 1972. (document), 2.1, 2.2.1

[Tos] Toshiba Corporation, *Bee-gent*, http://www2.toshiba.co.jp/beegent/index.htm. 1.2.1

[Tro] *Tropos*, http://www.troposproject.org/. 1.2.1

[Try04] Tryllian, *Adk 3.0*, http://www.tryllian.com/technology/product1.html, 2004. 1.2.1

[Vel97] J. Velasquez, *Modeling emotions and other motivations in synthetic agents*, 1997. 7.2.1, 7.2.2, 7.2.2

[VPF98] Rodrigo M. M. Ventura and Carlos A. Pinto-Ferreira, *Emotion-based agents*, AAAI/IAAI, 1998, p. 1204. 7.2.1, 7.2.2

[WH97] R. C. Wilson and E. R. Hancock, *Structural matching by discrete relaxation*, IEEE Transactions on pattern analysis and machine intelligence **19** (1997), no. 6, 634–698. 3.4.2, 5.6.2

[WJK00] Michael Wooldridge, Nicholas R. Jennings, and David Kinny, *The gaia methodology for agent-oriented analysis and design*, Autonomous agents and multi-agent systems **3** (2000), no. 3, 285–312. (document), 1.2.1, 1.2.1

[Woo00] Wooldridge, *Reasoning about rational agents*, The MIT Press, Cambridge, Massachussetts, 2000. 1.2.2

[Zal02] Michal Zalewski, *Strange attractors and tcp/ip sequence number analysis - one year later*, http://lcamtuf.coredump.cx/newtcp/, 2002. (document), 2.2.4, 2.2.10, 2.2.4

[Zei76] B. Zeigler, *Theory of modelling and simulation*, Robert E. Krieger Publishing Company, Malabar, Fl, USA., 1976. 1.2.2

Application 3

7.4 Introduction

Les travaux de la thèse présentés jusqu'à présent ont donné suite au développement d'une troisième application.

Ces travaux sont en cours et il s'agit d'une collaboration avec Mickael Camus, un doctorant de Paris 6 sous la direction d'Alain Cardon. Frank Chevallereau, Geoffrey Plouviez et Nicolas Vasquez, élèves à l'Epitech, sont également intervenus sur le projet sous la responsabilité de Mickael Camus pour développer des modules du système, notamment des parties proches de la couche matériel robot (interface robot, saisie des informations, reconnaissance d'images...).

Par ailleurs, ces travaux seront poursuivis dans un cadre industriel par la suite.

7.5 Description Générale

Le but de ce projet est de construire un système de pilotage pouvant se greffer sur tout type de robot pour conduire son comportement. En particulier, pour notre développement actuel, nous utilisons des robots de type Aibo.

Le projet actuel comprend une partie de portage du prototype présenté précédemment, en deuxième application initialement développé en O'Caml, dans un nouveau langage, le Oz (`www.mozart-oz.org`).

Le système développé est schématisé sur la figure 7.5.1. Les robots Aibo sont pilotés par le système qui est réparti sur une grappe de machines faisant tourner des machines virtuelles Oz/Mozart. Le système pilote le comportement des robots par un serveur d'interface.

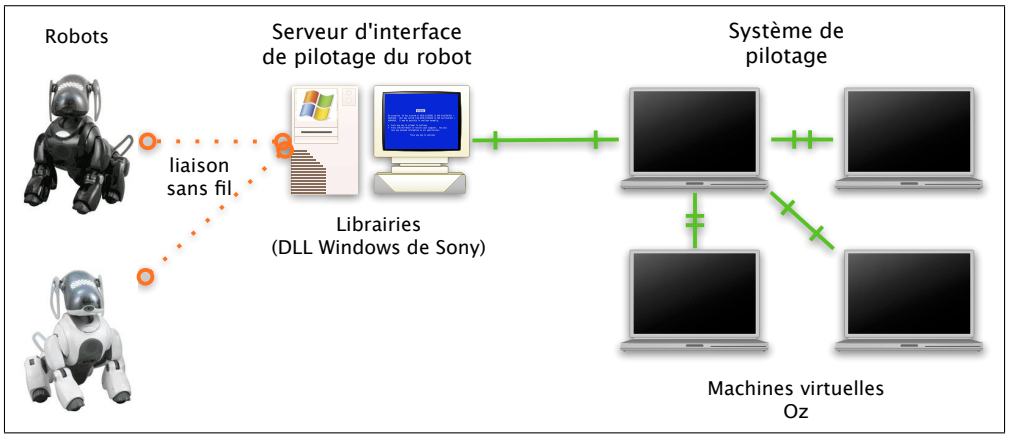


Figure 7.5.1 – *Boucle de pilotage des robots.*

7.6 Oz – Mozart

Mozart est un système de programmation pour applications distribuées. Mozart est fondé sur le langage Oz, langage multi-paradigmes : programmation déclarative, programmation par objets, programmation par contraintes, la notion de concurrence est totalement cohérente dans le langage, de plus, pour la distribution, le langage fourni une couche réseau transparente.

La machine virtuelle Oz-Mozart est disponible sur les plates-formes Unix (Linux, FreeBSD, Mac OS X…), sur Windows…

7.7 Description des robots utilisés

Les robots sur lesquels nous travaillons sont des robots ERS-7 de Sony, ou plus connus sous le nom de « Aibo ».Ces robots ont l'apparence d'un chien (voir image 7.7.1).

Figure 7.7.1 – *Photo d'un Aibo ERS-7 noir.*

Les deux robots disponibles sur lesquels nous travaillons sont fournis par :

- le laboratoire Leria (Aibo de couleur noir) ;
- et l'association Automates intelligents (Aibo de couleur blanche).

Le robot Aibo possède deux modes :

- un mode dit « autonome » où l'utilisateur interagit avec le robot comme un jouet. Le comportement du robot n'est pas directement commandé par l'utilisateur mais il peut être modulé par des interactions, comme par exemple des « caresses » donnée au robot, pour renforcer tel ou tel comportement, ou des ordres oraux, des mots que le robot peut reconnaître ;
- un mode commande :
  - il est possible de piloter ces robots par l'interface commerciale fournie par Sony,

– ou bien de développer son propre système en s'appuyant sur différentes librairies telles que, par exemple Open-R, Tekotsu.

7.8 Interface robot–système

Pour piloter les robots, nous avons choisi de télécommander le comportement du robot par notre système. Il ne s'agit pas d'un système de commande classique à proprement parler. Le système de pilotage n'est pas embarqué sur le robot même car les ressources sont insuffisantes sur le robot en terme de mémoire et de puissance de calcul. Le système de pilotage tourne sur des machines plus puissantes et commande le robot par le biais d'une interface qui fonctionne sur un serveur dédié.

Ce serveur tourne sur une plate-forme Windows car il exploite des librairies (DLL) de Sony publiquement disponible uniquement sur Windows. Ces librairies permettent l'accès à un répertoire de commande d'un certain niveau d'abstractions sans avoir à programmer et à calculer les trajectoires de chaque membre du robot, tels que : bouger la tête de tant et tant degrés, avancer, reculer, tourner à gauche ou à droite, se coucher, se lever...

7.8.1 Connexions

Les connexions aux robots se font par TCP/IP sur liaison sans fil « WiFi » (802.11g) en réseau local fermé. Le serveur d'interface se connecte au robot par liaison sans fil. Le système de pilotage se connecte ensuite au serveur d'interface par liaison conventionnelle (ethernet ou même sans fil).

L'interface a été conçue de façon à pouvoir gérer plusieurs robots en même temps. Nous utilisons en l'occurrence, pour l'instant deux robots Aibo.

7.8.2 Données

L'état de l'ensemble des capteurs est rafraîchi régulièrement. Ces données sont récupérées par le serveur interface. Ces données sont ensuite transcrites sous forme symboliques et mise à disposition du système de pilotage.

Capteurs

Le robot possède un certain nombre de capteurs :

- caméra couleur ;
- microphone stéréo ;
- capteurs de toucher, sur la tête, sur le dos ;
- capteurs de distance (deux : distances « courtes », distances «longues») ;
- les servomoteurs, ce sont plutôt des actionneurs, mais il est possible de connaître les angles de rotations ;
- des accéléromètres, au nombre de trois, permet de connaître l'inclinaison du robot selon les trois directions de l'espace.

Effecteurs

Le robot possède également un certain nombre d'effecteurs, lui permettant d'évoluer dans l'espace et de communiquer :

- lumière : diodes placées sur la tête du robot, le visage, sur le dos ;
- son : un haut parleur qui permet au robot d'émettre des sons ou de la musique ;
- les servomoteurs, ce sont plutôt des actionneurs, mais il est possible de connaître les angles de rotations et donc de s'en servir également comme capteurs.

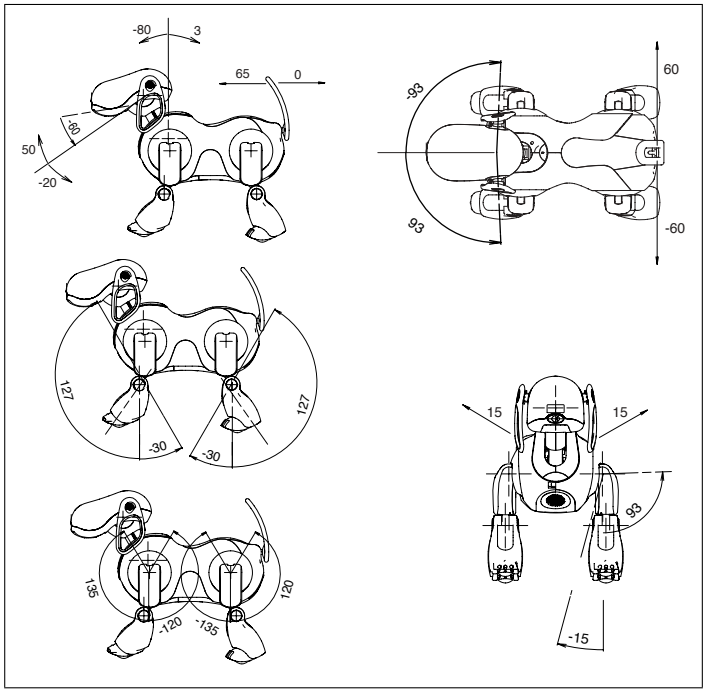


Figure 7.8.1 – *Mouvements possibles des membres de l'Aibo (unité : degré).*

7.9 Système de conduite comportementale

Le comportement des robots est piloté par un système basée sur des organisations d'agents. À l'instar de la deuxième application développée dans cette thèse, une organisation d'agents aspectuels construit une représentation de la scène courante du robot. Cette organisation est analysée par une organisation morphologique qui décrit l'état de l'organisation aspectuelle qui est ensuite pilotée par une organisation d'analyse.

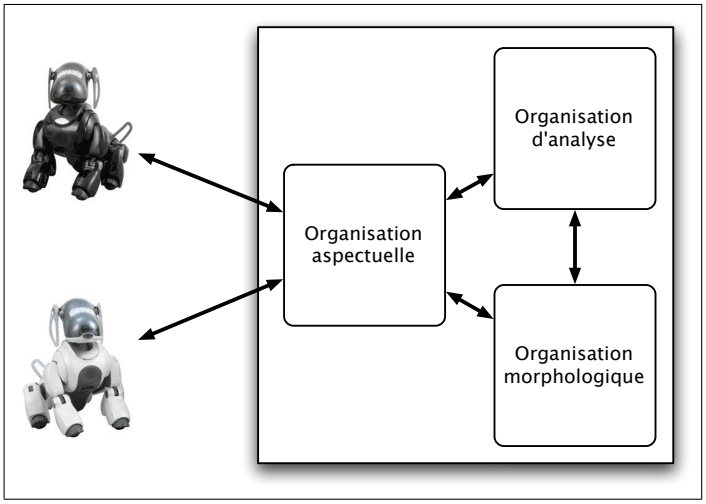


Figure 7.9.1 – *Boucle de pilotage des robots par les organisation d'agents.*

7.9.1 Agents

Le portage en Oz du prototype développé en O'Caml reprend en grande partie les spécifications initiales de la deuxième application. Cependant le langage Oz possède des caractéristiques rendant très facile l'implantation de certaines spécificités, telles l'asynchronisme et la distribution.

Dans ce portage, les agents ne sont plus synchrones comme dans le prototype de la deuxième application. L'activité du rôle se décline en deux temps :

1. le filtrage des données,
2. le calcul effectif des objets à émettre.

Néanmoins pour ne pas que les agents réagissent tout de suite au moindre message qui transite dans l'organisation, nous avons temporisé l'activation du rôle de l'agent.

Techniquement cela est fait en lançant deux fils d'exécution, un qui se charge du filtrage et le second qui se charge du calcul effectif. La récupération des données intéressant le rôle se fait dans une mémoire tampon, à exclusion mutuelle pour ces deux processus (voir figure 7.9.2).

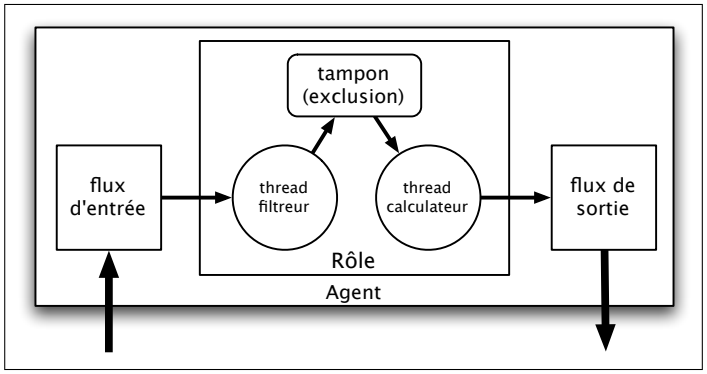


Figure 7.9.2 – *Architecture du rôle.*

Le fil d'exécution de filtrage s'active automatiquement quand une donnée arrive dans le flux d'entrée. Le second fil d'exécution, celui du calcul, s'active de façon périodique à une fréquence donnée et modifiable à la volée.

7.9.2 Agents d'interface

Les agents d'interface récupèrent les informations des capteurs issus du serveur d'interface et les mettent à disposition des autres agents aspectuels.

Par ailleurs, un agent spécifique dit « agent de commande », a pour rôle de faire une synthèse de tous les ordres de commande du robot issus des agents aspectuels sur un temps donné pour envoyer ensuite une commande « raisonnable » au robot, par exemple pour éviter des ordres contradictoires comme avancer et reculer en même temps, la prise en compte des ordres contradictoires est faite au niveau de cet agent.

7.9.3 Agents aspectuels

L'organisation aspectuelle fonctionne en grande partie comme cela a été présenté dans l'application deux de la thèse. Le cœur du système de pilotage comportemental se trouve dans ces agents. Une ontologie est établie en premier lieu et ensuite les éléments sont réifiés dans les rôles des agents aspectuels.

Nous procédons en deux étapes en ce qui concerne le développement du comportement de ces agents :

- dans un premier temps les agents ont un comportement plutôt réactif (pas de notions de groupes explicite, par exemple) ;
- dans un second temps les agents auront davantage un comportement intentionnel et social (ajout de la notion de groupe explicite).

Nous avons choisi de procéder de cette façon pour réaliser le système en introduisant la complexité des agents de façon incrémentielle.

Certains agents aspectuels ont pour rôle de commander effectivement le robot en lui envoyant des ordres dédiés. Ces ordres sont en fait interceptés par l'agent spécifique, mentionné plus haut, qui en fait une synthèse avant d'envoyer un ordre au robot. D'autres agents seront davantage concernés par des notions plus détachées du robot.

7.9.4 Agents morphologiques

Les agents morphologiques ont pour rôle de procéder à l'analyse morphologique des vecteurs aspectuels des agents aspectuels.

Dans un premier temps, nous avons procédé à l'analyse des communications des agents. Les communications des agents sont analysées et représentées sous forme d'histogrammes (un peu comme dans la première application de la thèse). Ceci permettra de distinguer les agents selon leurs activités communicationnelles.

Dans un deuxièmement temps nous introduiront les proximités (comme dans la deuxième application de la thèse, analyse fondée sur les graphes) fondée sur la sémantique mais également sur le partage d'information.

Les agents morphologiques peuvent projeter la morphologie sur l'espace sémantique : la projection peut se faire sur un sous-espace de l'espace des rôles.

7.9.5 Agents d'analyse

Les agents d'analyse exploitent les descriptions morphologiques de l'organisation aspectuelle pour la piloter (comme cela a été fait dans la première et deuxième application).

7.9.6 Générateur de rôle

Succinctement, un rôle d'un agent peut être vu comme une fonction f produisant un résultat étant donnée un contexte donné $Y = f(X)$. Cette fonction est paramétrée par une série de variables a_i. Le générateur de rôle produit une série de rôles en donnant des valeurs spécifique à ces paramètres.

Considérons par exemple un rôle qui extrait les objets présents dans une zone de l'espace définie par X et Y et qui produit une translation définie par DX et DY. Le générateur de rôle prend alors en compte ces différents paramètres et le domaines de variations de ces paramètres (par exemple $X^- \ldots X^+$, $Y^- \ldots Y^+$, $DX^- \ldots DX^+$, $DY^- \ldots DY^+$) et génère un ensemble de points dans la boule définie par ces quatre domaines, au sens de la norme infinie (une boule « carrée »), chacun de ces points définissant un rôle particulier.

7.10 Divers

La figure 7.10.1 présente la fenêtre principale de l'interface visuelle du serveur interface entre le système et les robots.

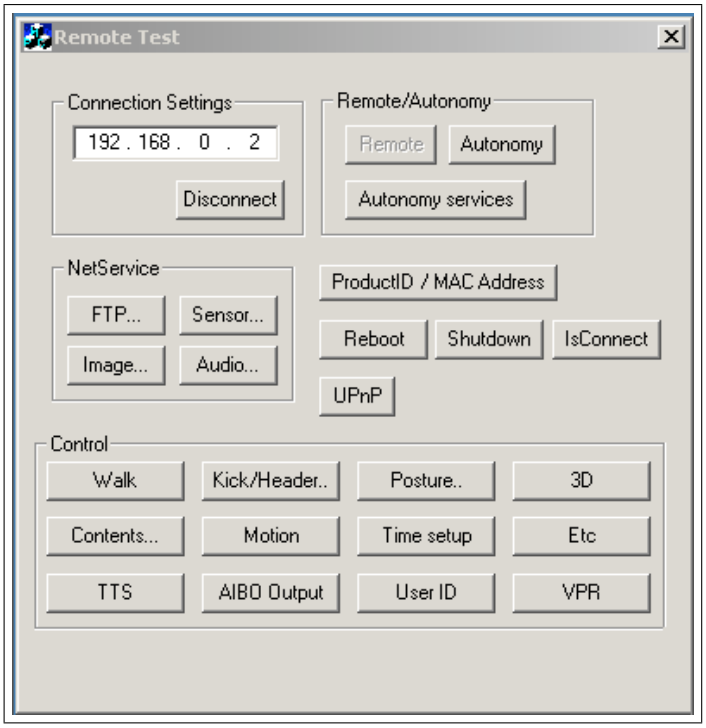


Figure 7.10.1 – *Capture d'écran de l'interface du serveur d'interface. Fenêtre principale.*

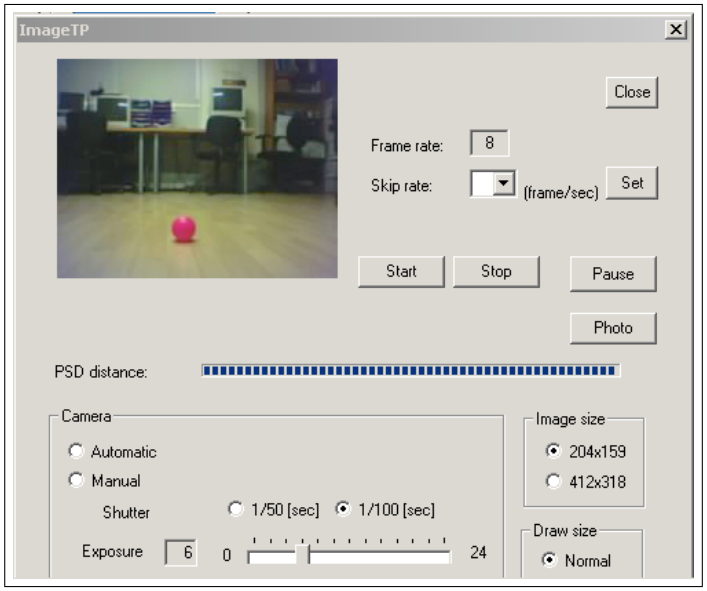


Figure 7.10.2 – *Capture d'écran de l'interface du serveur d'interface. Fenêtre de la caméra vidéo.*

La figure 7.10.2 illustre la boîte de dialogue du module de vidéo, nous pouvons, entre autre, apercevoir ce que « voit » l'Aibo. Nous pouvons apercevoir également la mesure du capteur de distance.

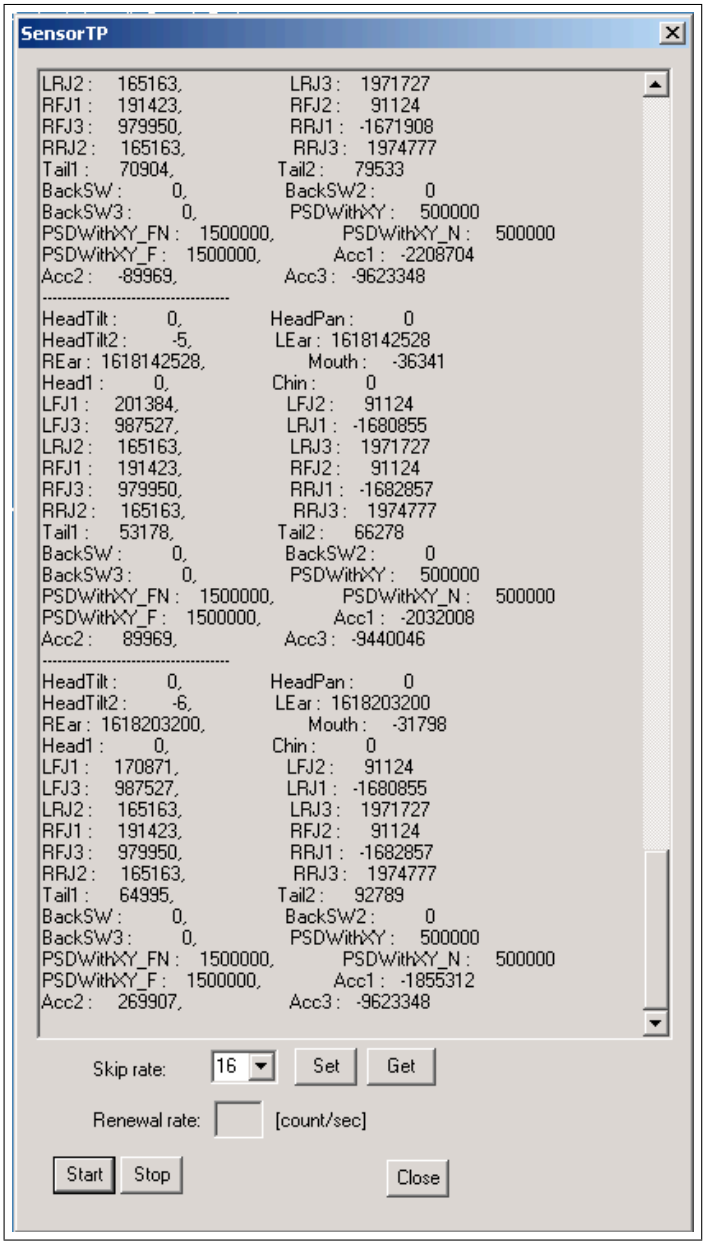


Figure 7.10.3 – *Capture d'écran de l'interface du serveur d'interface. Fenêtre des valeurs des capteurs.*

La figure 7.10.3 représente la boîte de dialogue affichant une série de prise d'information de l'Aibo. On peut y lire les valeurs de différents capteurs tels que, par exemple : les angles d'inclinaison de la tête, les angles des articulations,

l'orientation de la queue, les capteurs de toucher (des pattes, de la tête, du dos), les accéléromètres, les capteurs de distance.

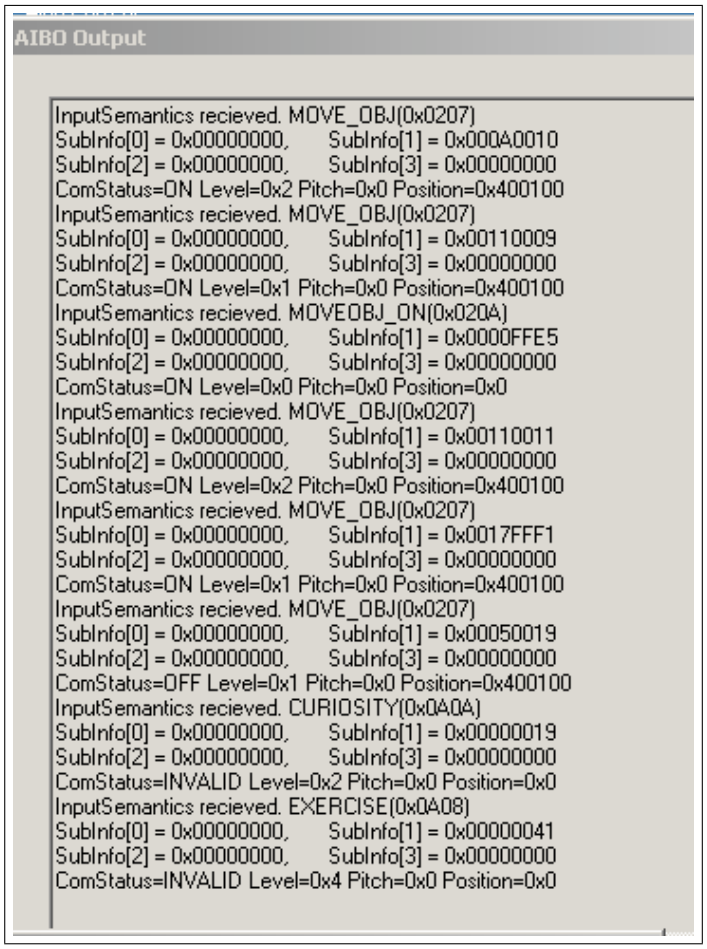


Figure 7.10.4 – *Capture d'écran de l'interface du serveur d'interface. Fenêtre des événements.*

Le figure 7.10.4 est un extrait du journal des événements tels que, par exemple, la détection d'un objet connu (comme la balle, l'os ou un autre objet appris par l'utilisateur), ou le mouvement d'un objet…

Nous expérimentons un système de visualisation en 3D, développé en s'appuyant sur OpenGL (voir figure 7.10.5). Dans cette première version, l'interface permet de visualiser le passage des messages entre les agents et leur intensité. Les agents sont regroupés selon leur rôles en boules. L'affichage se fait en temps

réel ici, contrairement à deuxième application de la thèse où il s'agissait d'une simulation.

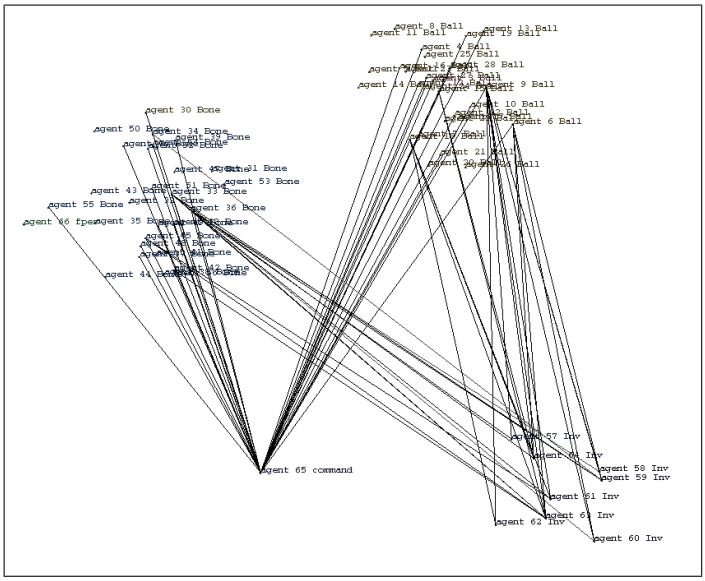


Figure 7.10.5 – *Capture d'écran du système de visualisation en 3D.*

7.11 Conclusion

La boucle de contrôle est implantée dans une première version. Nous travaillons actuellement sur un exemple d'ontologie à réifier dans le système et nous envisageons d'augmenter fortement le système par la suite (ajout de fonctionnalités au niveau du comportement des agents, augmentation de l'interface homme-machine). Nous prévoyons de faire ceci dans le cadre d'un projet industriel.