# Démarche, modèles et outils multi-agents pour l'ingénierie des collectifs cyber-physiques

Jean-Paul Jamont

UNIVERSITÉ GRENOBLE ALPES

# MÉMOIRE

Présenté et soutenu publiquement pour l'obtention de l'

## HABILITATION À DIRIGER DES RECHERCHES DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Informatique**

Présenté par

**Jean-Paul Jamont**

# Démarche, modèles et outils multi-agents pour l'ingénierie des collectifs cyber-physiques

Soutenue publiquement le **10/02/2016**,
devant le jury composé de :

**M. Olivier Boissier** Professeur, Président
**Mme Marie-Pierre Gleizes** Professeur, Rapporteur
**M. Abderrafiâa Koukam** Professeur, Rapporteur
**M. Michel Occello** Professeur, Examinateur
**M.Eugénio C. Oliveira** Professeur, Rapporteur
**Mme Florence Maraninchi** Professeur, Examinateur

À la mémoire de ma mère,
à Capucine et Alice.

# Remerciements

Que les membres du jury trouvent dans ces quelques lignes l'expression de mes plus sincères remerciements pour le temps qu'ils m'ont consacré ainsi que pour toutes leurs remarques qui m'aideront tout au long de mon projet. Merci à Marie-Pierre Gleizes, Abderrafiâa Koukam et Eugénio C. Oliveira d'avoir rapporté sur ce travail. Merci à Olivier Boissier et Florence Maraninchi de m'avoir fait l'honneur de participer à ce jury.

Que Choukri-Bey Ben-Yellès, auprès duquel j'apprends beaucoup, trouve aussi dans ces remerciements l'expression de toute ma reconnaissance.

Le travail présenté dans cette habilitation est le résultat de nombreuses expériences partagées. Aussi, c'est à Michel Occello qui m'a accompagné dans la plupart d'entre elles que je souhaite adresser en premier lieu mes plus chaleureux remerciements : tu as non seulement très amplement marqué ma perception, ma compréhension et ma vision du domaine multi-agent mais tu m'as aussi formé au métier d'enseignant-chercheur. Merci d'avoir toujours été présent, tant dans mes moments de certitudes que de doutes.

Les contributions de ce mémoire sont des productions collectives. Je remercie l'ensemble des étudiants en thèse que j'ai eu le plaisir d'encadrer. Merci à Francisco Cervantès, Nacer Hamani, Thi-Thanh-Ha Hoang, Ndiaye Khadim, El Mehdi Khalfi, Afra Khenifar. Merci également aux étudiants de master et de DRT.

Ce travail a aussi directement profité de l'influence d'une active communauté française de recherche en Systèmes Multi-Agents. Qu'elle trouve ici l'expression de mes remerciements. J'ai une pensée toute particulière pour Flavien Balbo, Laurent Vercouter, Nicolas Sabouret (bonne année !), Sébastien Picault, Gauthier Picard, Rémy Courdier, Philippe Mathieu, René Mandiau et Yves Demazeau.

Je remercie aussi les membres du laboratoire et en particulier ceux de l'équipe Macsy-Cosy que je n'ai pas déjà cités : Laurent Lefèvre qui dirige notre équipe mais aussi Damien Genthial, Jean-Luc Koning, André Lagrèze, Eduardo Mendes, Annabelle Mercier, Ionela Prodan et Clément Raïevsky. Merci aussi à Jennyfer Duberville et Carole Seyvet pour leur soutien.

Je souhaite également remercier des collègues avec lesquels j'ai eu particulièrement plaisir à travailler : Cécile Favre, Virginie Fresse, Mouloud Koudil (avec une pensée à la mémoire de Karim Benabadji), Lionel Médini, Mickael Mrissa, Frédéric Pourraz et Nicolas Stouls.

Je remercie Roland Pelurson, directeur de l'IUT de Valence, et Thierry Vincent, directeur-adjoint, pour leur soutien dans mes activités de recherche et notamment dans l'organisation des JFSMA à Valence.

Je tiens aussi exprimer mes amitiés à mes collègues de l'IUT et à remercier Christian Duccini, Nicolas Fourty et Christine Galdon du département Réseaux et Télécommunications : grâce à votre implication dans le département, vous m'avez permis de travailler plus sereinement sur ce manuscrit. Je tiens aussi à distinguer Cécile Metge pour les relectures de mes articles anglais.

Merci à ma famille et à mes amis (mention spéciale pour Laurent) pour leur intérêt et leur soutien. Capucine et Alice, sans vous rien de tout ça n'aurait réellement de sens.

# Table des matières

# Table des figures

# Liste des tableaux

# Chapitre 1

# Introduction

Ce manuscrit présente une synthèse des travaux de recherche que j'ai effectués de 2005 à 2015 en tant que Maître de Conférences à l'Université Pierre Mendès France (UPMF) (Grenoble 2), enseignant à l'IUT de Valence et chercheur au Laboratoire de Conception et d'Intégration des Systèmes (LCIS) au sein de l'équipe COSY. Ma thèse soutenue en 2005, proposait un cadre méthodologique pour la construction des systèmes multi-agents embarqués. Depuis, je développe démarches, modèles et outils afin d'aider les concepteurs à construire ces systèmes complexes que sont devenus les systèmes embarqués.

## 1.1 Avènement des systèmes cyber-physiques

### 1.1.1 Environnement technique

**Les systèmes embarqués** Dans la plupart des définitions {Vahid and Givargis, 2002; White, 2011} des systèmes embarqués transparait un positionnement par rapport aux ordinateurs. D'aucuns les décrivent comme des objets mêlant logiciel et matériel dédiés à l'accomplissement d'une tâche précise par opposition à l'usage généraliste des ordinateurs. D'autres les comparent à des systèmes dans lesquels matériels et logiciels sont noyés de façon moins discernable que dans l'informatique traditionnelle et qui n'utilisent pas systématiquement des périphériques d'entrée/sortie classiques. C'est en fait dans l'importance cruciale que peuvent revêtir certaines contraintes non fonctionnelles qu'il faut chercher les éléments les différenciant : l'attention portée par les concepteurs à la sécurité des utilisateurs, à la gestion de l'énergie, à la recherche du faible coût de production unitaire etc. Ces contraintes imposent souvent l'utilisation de micro-processeurs ou de micro-contrôleurs peu coûteux disposant de capacités de stockage limitées que ce soit pour le code ou pour les données.

**Les systèmes embarqués distribués** Avec l'évolution des moyens de communication sans fil, les systèmes embarqués jusqu'alors généralement monolithiques et de petite échelle sont devenus de plus en plus distribués et communicants {Kopetz, 2011}. En effet, même si les communications sans fil analogiques existent depuis de nombreuses années, leur développement a connu un renouveau en 1997 avec les premières contributions des groupes de travail 802.11 de l'Institute of Electrical and Electronics Engineers (IEEE) et Mobile Ad hoc NETwork (MANET) de l'Internet Engineering Task

Force (IETF). C'est ainsi que se développa l'informatique ambiante {Zelkha et al., 1998} et qu'un second souffle fut donné à l'informatique ubiquitaire, le Machine-to-machine (M2M) etc.

**Les systèmes cyber-physiques** Dans de plus en plus d'applications, le monde virtuel impacte le monde physique à l'aide d'effecteurs et, réciproquement, les grandeurs du monde physique mesurées par des capteurs impactent les modèles et les comportements virtuels. On qualifie souvent de tels systèmes embarqués de cyber-physiques {Lee, 2008}.

**L'internet des objets** L'International Telecommunication Union (ITU) s'est intéressé à l'intégration des objets communicants à l'Internet ITU {2005} c'est-à-dire des systèmes embarqués au monde d'Internet Protocol (IP). Ces objets peuvent désormais interagir entre eux ou avec des logiciels comme les services Web. Au-dessus de l'infrastructure que représente l'IoT, a émergé le Web of Things (WoT) {Raggett, 2010} qui s'intéresse aux fonctionnalités que proposent ces objets c'est-à-dire aux services qui ont un sens pour l'utilisateur.

### 1.1.2 Environnement social

Les objets cyber-physiques sont présents dans notre environnement. Ils parlent entre eux et notamment de nous, les humains de plus en plus connectés. Nous inventons de nouveaux termes pour parler de ces objets si bavards (blogject, tweetject, spim etc.).

**Les objets sont partout** Chaque année, de nouveaux objets apparaissent et participent à créer autant de nouveaux usages. Par exemple, l'entreprise française Withings a mis sur le marché des pèse-personnes intelligents connectés fournis avec une application permettant de jouer le rôle de coach pour atteindre des objectifs de poids. Vitality est une entreprise américaine qui propose *GlowCap*, un semainier à destination des personnes fragiles capable de signaler à l'utilisateur qu'il oublie de prendre ses médicaments, de générer des rapports de consommation pour le médecin et de prévenir le pharmacien lorsqu'il est nécessaire de se réapprovisionner. Le *SnifTag* est un collier pour chien qui récolte des données sur le bien-être de l'animal. *Botanicalls* permet à une plante d'envoyer un tweet à son propriétaire si elle manque d'eau. Siemens propose un système d'étiquettes virtuelles permettant, lorsqu'on se rend sur un lieu touristique, de laisser des messages pour ses amis qui repasseront par ce lieu. Ces objets proposent de nouveaux usages et il est possible de les détourner de leur fonction première afin d'en créer de nouveaux.

**Les objets parlent entre eux** Un objet communique avec un autre pour de multiples raisons : il souhaite mettre à jour ses primitives de service (une télévision connectée qui met à jour ses codecs), combler un manque de connaissance ou de service (un système d'arrosage qui interagit avec une station météo pour savoir s'il va pleuvoir, un thermostat qui utilise un radiateur introduit dans la même pièce que lui afin d'asservir la température) ou pour acheminer une information vers un tiers qu'il ne peut directement atteindre.

**Les objets parlent de nous** Les objets cherchent globalement à atteindre la satisfaction de l'utilisateur (artificiel ou humain). Pour cela, ils échangent les objectifs exposés par l'utilisateur voire son profil. Les profils contiennent de nombreuses informations qui agissent comme des préférences de l'utilisateur et qui peuvent impacter le comportement d'un objet ou sa manière d'interagir. Afin de faire correspondre au mieux les services aux attentes des utilisateurs, ils peuvent contenir des

informations sur leurs états civils ou même leurs historiques d'interactions.

**On parle d'eux** Avec l'arrivée des objets connectés dans son environnement, de nouveaux termes ont été créés. On parle par exemple de *things' spamming* pour parler des communications électroniques non désirées ou non sollicitées mises en œuvre par les objets. Les termes *blogject* et *tweetject* désignent un objet capable respectivement de publier régulièrement des informations sur des sites Web ou d'envoyer des messages via la plateforme Twitter. La notion d'*informational shadow* qu'introduit Mike Kuniavsky fait référence à l'information qui est associée à un objet comme son nom, son numéro, sa position dans l'espace et le temps, et ainsi de suite. Il s'agit aussi des métaphores qui aident les gens à comprendre les nouveaux services en les reliant à des choses familières. Bruce Sterling introduit le terme *spime*. Il s'agit d'un néologisme pour désigner un objet théorique qui peut être suivi à travers l'espace et le temps tout au long de sa durée de vie. Un *physical mashup* désigne une application qui combine du contenu et des services provenant d'objets connectés hétérogènes.

## 1.2 Contributions à l'ingénierie des collectifs cyber-physiques

### 1.2.1 Une approche guidée par le paradigme multi-agent

**Une approche nécessairement décentralisée** Plusieurs critères militent pour la décentralisation de l'intelligence : tout d'abord ces systèmes embarqués sont par nature physiquement distribués. Ensuite, ces systèmes doivent être ouverts, c'est-à-dire qu'ils ont à supporter l'ajout d'objets ou leur départ, ce qui inclut les services et les données qu'ils portent. Plus loin, ils doivent être capables de s'adapter à l'ajout massif d'objets (problème du passage à l'échelle). Raisonner à partir d'un modèle complet du système est difficile car, d'une part, sa construction et sa maintenance demanderaient de nombreuses ressources et, d'autre part, une fois obtenu ce modèle ne reflèterait plus forcément l'état du système. Enfin, un même objet peut participer simultanément à plusieurs applications : de nombreux conflits peuvent alors se produire à différents niveaux du système global. Un objet doit donc être autonome et proactif c'est-à-dire qu'il doit avoir le contrôle de son état et de son comportement afin notamment de s'adapter aux évolutions de son environnement. Il doit décider d'initier des interactions locales/globales avec les autres objets en fonction des objectifs locaux/globaux des différentes organisations auxquelles il appartient. La conception de ces systèmes doit donc suivre une approche ascendante : à partir des compétences et connaissances identifiées localement, il convient d'essayer de répondre au mieux aux attentes d'utilisateurs (artificiels ou humains). En d'autres termes, un objectif de niveau global ou une mission est pris en charge par un objet, le collectif lui permettra de trouver les compétences et les connaissances qui lui font défaut. Chacun des objets respecte des considérations d'ordre stratégique qu'elles soient communes ou individuelles : politique de gestion de l'énergie, de gestion des données privées/sensibles etc. Il faudra alors que l'objet responsable d'une mission négocie avec les autres objets ou qu'il entre éventuellement dans des démarches explicatives pour convaincre un autre objet de changer sa consigne (modèles d'argumentations).

**Une modélisation multi-agent** Dans le contexte des systèmes cyber-physiques, je modélise les objets par des agents autonomes et je travaille à accommoder des comportements répondant à des besoins individuels des agents et des attentes du système global qui requièrent leur coopération.

Je m'intéresse donc plus particulièrement aux Collectifs Cyber-Physiques (CCP) et mes travaux s'inscrivent pleinement dans le domaine des Systèmes Multi-Agents (SMA). Les particularités des SMA qui m'intéressent découlent des propriétés de l'environnement physique. Cet environnement est en effet {Russel and Norvig, 1995} :

- inaccessible : les agents raisonnent à partir de vues et de compréhensions partielles de l'environnement (principe de localité) ;
- non déterministe : l'état futur de l'environnement ne dépend pas uniquement de son état courant et des actions effectuées des agents. Plus loin une perception est toujours liée à une incertitude de mesure et l'action d'un agent n'est pas exempt de défaillance ;
- séquentiel : la décision prise par un agent (ou un groupe d'agents) affecte ses décisions futures ;
- dynamique : lorsqu'un agent raisonne, l'état de son environnement évolue tout comme celui des autres agents ;
- continu : le potentiel en termes de primitives d'action et de perception possibles n'est pas aussi clairement limité que dans les environnements logiciels.

Un concepteur qui doit développer un SMA évoluant avec de tels environnements est souvent dans une situation peu confortable. En effet, s'il a une idée précise de l'objectif du système, il ne connaît aucun algorithme lui permettant d'y arriver, il ne peut pas inventorier toutes les situations que le système, et donc les agents, peuvent rencontrer et, en raison de la complexité des problèmes à résoudre, l'espace de recherche qu'il devrait explorer est important {Gleizes, 2004}.

### 1.2.2 Axes de recherche

La conception des collectifs cyber-physiques que je modélise avec des SMA nécessite une démarche méthodologique qui guide le concepteur du recueil des besoins jusqu'au déploiement du système et même à sa maintenance. De plus, des modèles de collectifs orientés individus compatibles avec les contraintes de ces systèmes sont nécessaires pour permettre de simplifier le travail du concepteur et assurer le maintien de l'intégrité fonctionnelle du système global, de permettre une compréhension de son fonctionnement, de gérer efficacement les communications, etc. Enfin, des outils doivent supporter la démarche et permettre la mise au point du système.

**Une démarche méthodologique** J'ai proposé durant ma thèse la méthode Decentralized Iterative Approach for Multiagent Open Networks Design (DIAMOND) qui permet de guider le concepteur d'un tel système complexe de son analyse à son implantation. Parce qu'une approche méthodologique doit être générique, avec mes collègues nous l'avons confrontée à de nombreux problèmes :

- industriels lors de projets de recherche appliquées ou de transfert technologique dans des domaines applicatifs relatifs à l'instrumentation sans fil, la robotique collective et l'assistance aux personnes dépendantes ;
- académiques lors de collaborations nationales et internationales sur des problèmes de commande de systèmes physiques, la réalisation de systèmes de géopositionnement ou d'information géographique.

C'est ainsi que nous avons été amenés à intégrer à la méthode une phase de qualité, à développer

le recueil des exigences extra fonctionnelles notamment relatives au déploiement des collectifs cyber-physiques et leur intégration dans les agents à l'aide du concept d'émotions etc.

**Des modèles de collectifs** Dans les applications qui nous intéressent, il est impossible de créer et de maintenir à coût raisonnable un modèle suffisamment complet et explicite du système à contrôler afin de permettre son contrôle total et centralisé. Les éléments qui composent le système doivent donc raisonner avec des descriptions partielles du système, décider de leurs actions en autonomie et s'organiser pour répondre aux objectifs globaux. Nous avons formalisé et validé des modèles orientés individus qui participent à faible coût (empreinte mémoire, consommation de temps de calcul) au maintien de l'intégrité fonctionnelle du système global. Lors de mes travaux de thèse, j'avais proposé Multi-Wireless-Agent Communication model (MWAC), un modèle d'auto-organisation générique que nous avons utilisé initialement pour gérer les communications dans des réseaux de capteurs. Depuis, toujours dans un souci de généricité, nous l'avons utilisé dans des contextes différents comme l'identification de ressources dans des réseaux sociaux. Pour faire face à des applications d'instrumentation critiques, nous avons développé un modèle léger de gestion de confiance que nous avons intégré à MWAC. Pour répondre à des contraintes de gestion de flux spécifiques, nous avons utilisé la propriété de diversification de l'optimisation par colonie de fourmis. Afin de rendre intelligibles des systèmes particulièrement étendus, nous avons intégré dans MWAC les mécanismes récursifs génériques pour le multi-niveau.

Nous avons par ailleurs mené d'autres travaux pour permettre de maintenir des compositions de services en environnements changeants. Ces derniers modèles ont été produits en nous inspirant des "écosystèmes".

**Des modèles d'agents** Les différentes facettes d'un SMA sont traditionnellement les Agents, l'Environnement, les Interactions et les Organisations (décomposition AEIO {Demazeau, 1995}). Dans le cas de SMA embarqués dans le monde réel, l'agent doit intégrer ces différentes facettes bien que ses ressources soient limitées (capacité de calcul, mémoire disponible pour le code et les données...). La nature physique de l'environnement nécessite qu'ils travaillent avec des données incertaines et qu'ils adaptent leurs comportements aux dynamiques temporelles des objets de l'environnement ainsi qu'à leurs possibles dysfonctionnements. Ils participent aussi à de nombreuses interactions logiques et physiques.

Pour répondre aux contraintes matérielles importantes que nous avions à respecter pour de nombreuses applications, nous avons spécialisé l'architecture d'agent ASTRO développée par Michel Occello en une architecture d'agent embarqué contrainte (architecture eASTRO). Dans le cadre d'un projet financé par l'Agence Nationale de la Recherche (ANR), nous avons construit une architecture d'agent permettant d'étendre les capacités et les connaissances d'objets physiques même inertes (objets sans capacité de calcul).

**Des outils** Il n'était pas suffisant de proposer une méthode et des architectures spécifiques d'agents embarqués. En effet, aux difficultés de la conception traditionnelle de SMA logiciels s'ajoutent les problèmes de déploiement sur des plateformes physiques, le test du SMA embarqué et son débogage. Le problème du passage à l'échelle pose celui de la mise au point d'un système embarqué qui nécessiterait d'avoir à disposition, tôt dans le cycle de développement, un nombre

important de plateformes. Dans ce contexte, nous avons proposé l'outil MultiAgent Software and Hardware simulator (MASH). À partir des besoins applicatifs et des performances attendues du système à concevoir, on construit un SMA à l'aide de la méthode DIAMOND. On simule alors le SMA et on l'ajuste afin que les résultats mesurés soient en phase avec les besoins exprimés. Une fois cette étape terminée, on va pouvoir s'intéresser à l'implantation du SMA sur une plateforme spécifique (cas du simple déploiement) ou construire la partie matérielle de l'agent. Le simulateur va accompagner l'embarquement du SMA en assurant une gestion du risque. Le risque majeur que l'on peut rencontrer est l'introduction de déviations dans le fonctionnement global du SMA. Elles proviennent de la dégradation des modèles originels mis en œuvre dans le SMA pour prendre en compte les différentes contraintes sur les ressources disponibles, sur les aspects temps réel, sur l'ergonomie du système, sur sa consommation énergétique etc.

Ces travaux ont bénéficié du soutien de l'équipe COSY et notamment de Michel Occello, avec qui j'ai encadré la quasi-intégralité des étudiants de thèse qui ont participé au projet de recherche dont je rends ici compte. Ces contributions sont aussi le produit de nombreuses collaborations nationales ou internationales, qu'elles soient académiques ou industrielles. Elles ont aussi subi l'influence de la très dynamique communauté SMA française.

## 1.3 Organisation du manuscrit

Le second chapitre de ce manuscrit présente une synthèse des nombreux défis et enjeux de la conception des CCP. Ils me permettront de préciser pour chaque contribution les challenges relevés, les travaux existants et les limites de nos propositions.

Le troisième chapitre présente la synthèse des questionnements que nous avons partagés sur les aspects méthodologiques de la conception des CCP. Ces questions concernent notamment le cycle de vie, le recueil des besoins, les phases de conception et d'implantation.

Le quatrième chapitre fait le point sur les modèles dédiés collectifs que nous avons développés. Ces modèles sont centrés individus notamment en raison de l'importance, dans un contexte embarqué, de leur intégration dans les architectures d'agents. Deux architectures d'agents et des modèles pour le collectif sont présentés.

Le cinquième chapitre se focalise sur l'outil MASH que nous proposons afin de supporter notre méthode de la conception du système à son déploiement dans le monde physique. Un cycle de vie original de la simulation des CCP est alors proposé.

# Chapitre 2

# Défis et enjeux de la conception des collectifs cyber-physiques

Nous présentons dans ce chapitre les défis que relève un concepteur de collectifs cyber-physiques (CCP). Pouvant être considérés comme un cas particulier de système embarqué, nous mettons en évidence cette filiation des défis de conception. Les défis présentés proviennent d'une synthèse de différents travaux du domaine {Lee, 2002; Elmenreich, 2003; Henzinger and Sifakis, 2006; Lee, 2008; Pottie and Kaiser, 2009; Zurawski, 2009} et de notre propre expérience dans la conception des systèmes embarqués {Jamont and Occello, 2016}.
Nous proposons aussi quelques définitions dont certaines ont été esquissées en introduction.

## 2.1 Des logiciels et des systèmes embarqués

Un CCP est avant tout un système embarqué. De nombreuses définitions des systèmes embarqués s'attachent en priorité à leurs spécificités physiques {Heath, 1997; Vahid and Givargis, 2002; White, 2011}. L'importance que revêt la prise en compte des contraintes non fonctionnelles nous amène à adopter la définition suivante :

**Définition: Système embarqué**

Un système embarqué est composé d'au moins un logiciel "embarqué" ou "intégré" dans une partie matérielle aux ressources limitées. Il est dédié à l'accomplissement de tâches pré-établies et spécialisées (par opposition à l'usage généraliste des ordinateurs). Plongé dans un monde physique qu'il peut partager avec l'humain, sa conception nécessite de porter une attention toute particulière au recueil des exigences non fonctionnelles et à leur intégration dans les agents.

Tout ou partie de la partie décisionnelle des systèmes embarqués peut faire l'objet d'une implantation logicielle ou matérielle. Si la flexibilité et la complexité algorithmique des traitements militent clairement pour des implantations logicielles, le temps de réponse, l'empreinte mémoire, la consommation énergétique, la fiabilité de fonctionnement militent fortement pour les implantations matérielles. On désigne sous cette dernière dénomination la mise en relation de portes logiques et autres connexions des composants électroniques utilisés (Programmable Array Logic (PAL), Erasable Programmable Logic Device (EPLD), Field-Programmable Gate Array (FPGA), etc.). Lors d'une implantation logicielle, un microprocesseur exécute des instructions préalablement stockées en mémoire (Read Only Memory (ROM) ou Random Access Memory (RAM)).

Les logiciels embarqués ne doivent pas seulement être vus comme des logiciels conçus pour des "petits ordinateurs". Ils adressent en plus tout un ensemble de contraintes extra-fonctionnelles et notamment celles liées à la sécurité des biens et personnes.

Les systèmes embarqués doivent respecter des impératifs pour répondre à leurs objectifs qui sont tout autant de défis pour le concepteur :

**Défi 1. Réactivité (*Reactivity*)** En raison de leur forte relation avec le monde physique, les systèmes embarqués doivent être particulièrement attentifs à leur environnement afin de réagir de façon appropriée aux événements qu'ils détectent. Que ce soit pour maintenir la qualité du service qu'il rend ou sa propre intégrité fonctionnelle, le système embarqué doit être capable de prendre des décisions de façon dynamique. Le défi consiste alors à réagir simultanément à de multiples stimuli et à modifier son comportement (ou même sa structure) sans nécessiter d'arrêt, de recompilation ou rupture de service.
Par exemple, quand un robot exécute un plan pour fermer une porte, il doit réagir à sa fermeture par un autre robot : l'achèvement du plan n'est plus nécessaire.

**Défi 2. Terminaison (*Liveness*)** L'informatique embarquée ne peut pas accepter la fin prématurée de programmes : il n'est pas suffisant qu'une bonne décision soit prise à partir des événements détectés. En effet, les blocages doivent à tout prix être évités notamment en raison des conséquences nuisibles qui peuvent survenir (violation de la sécurité des biens et personnes). Le système doit donc être capable d'introspection afin d'être sûr de son bon état de fonctionnement. Le cas échéant, il doit être en mesure d'adapter son comportement pour que la réponse soit satisfaisante, même si ce n'est que partiellement.
Typiquement une mission d'interception impose de réagir dans une certaine fenêtre temporelle.

**Défi 3. Ponctualité (*Timeliness*)** Parce que les processus physiques avec lequel il interagit évoluent au fil du temps, un système embarqué ne doit pas uniquement être réactif : il doit être ponctuel. Le temps de réponse [1] du système doit être compatible avec la dynamique du système physique. Réactivité et ponctualité garantissent le respect des contraintes temps réel. Le défi pour les concepteurs de logiciels embarqués est d'offrir suffisamment d'abstraction de contrôle du temps pour garantir que le logiciel va évoluer en accord avec la dynamique de l'environnement et terminer ses actions dans les temps. Traiter avec le temps est notamment crucial pour la sécurité globale du système.

1. Temps écoulé entre l'instant où le système décide d'agir et l'instant où il a satisfait l'objectif visé.

Un robot mobile doit être capable de générer à temps des chemins alternatifs pour éviter des obstacles, même si le trajet d'évitement adopté n'est pas le meilleur.

**Défi 4. Autonomie d'énergie (*Power Autonomy*)** Les logiciels embarqués sont souvent déployés sur de petits dispositifs mobiles pour lesquels l'approvisionnement en énergie n'est pas simple. Pour rester opérationnel le plus longtemps possible, le processus de décision doit gérer les tâches en fonction de leur importance et des ressources énergétiques disponibles. Le défi est donc d'intégrer la contrainte énergétique au sein du cycle de décision du système embarqué qui peut ainsi être dans des états de pleine capacité (efficacité de la réponse apportée aux besoins fonctionnels) ou des états de sommeil (réponse dégradée aux besoins fonctionnels mais économie de l'énergie).
Un drone immobilisé au sol doit trouver un bon compromis entre envoyer fréquemment sa position (pour être sûr qu'un véhicule de recherche qui passe dans son voisinage le détecte) ou espacer les envois pour être détectable pendant une plus longue période de temps.

**Défi 5. Gestion de la sécurité (*Safety management*)** Comme ils sont fortement liés aux environnements physiques, les systèmes embarqués peuvent exécuter des actions dangereuses pour l'homme, pour d'autres systèmes ou pour eux-mêmes. Leveson {2004} soutient que les accidents dans les systèmes complexes se posent en raison des interactions homme/machine mal comprises ou anormales. Le défi consiste à identifier et prendre en compte ces risques à la fois dans la phase de conception du système et lors de son fonctionnement.
Si un arrêt d'urgence est déclenché, un bras robotisé qui lève une importante charge doit tout de même resté en position afin de ne pas blesser un opérateur humain qui serait éventuellement en dessous de la charge.

## 2.2 De la mise en réseau des systèmes embarqués

Détaché d'un domaine applicatif particulier, le dictionnaire de l'académie française définit la coordination comme *l'action d'agencer certaines choses entre elles suivant les rapports qu'elles doivent ou peuvent avoir, les disposer convenablement pour une fin.* En informatique, les travaux abordant la coordination se réfèrent souvent à deux publications proposées par Malone and Crowston {1990, 1994} qui définissent la coordination ainsi : *the act of managing interdependencies between activities performed to achieve a goal.* Dans le domaine des SMA, la définition de N. R. Jennings est généralement rappelée {Jennings, 1996} : *the process by which an agent reasons about its local actions and the (anticipated) actions of others to try and ensure the community acts in a coherent manner.*

Nous appellerons dans la suite "coordination" l'action d'ordonner des actions.

**Définition: Coordination**

La coordination consiste à définir un agencement d'activités indépendantes qui ont préalablement été réparties afin de répondre à un objectif global. La coordination est généralement centralisée sur un nœud qui dispose du schéma de résolution du problème. Ce coordinateur organise l'agencement des tâches et suit son exécution.

Cette coordination est l'élément qui a motivé l'apparition des Systèmes Embarqués en Réseaux et les différencie des systèmes embarqués traditionnels {Zurawski, 2009}.

**Définition: Système embarqué en réseau (SER)**

Un système embarqué en réseau est un système dont la fonctionnalité est assurée en coordonnant les capacités de perception et d'action de différents sous-systèmes embarqués formant le réseau.

La coopération, tout comme la coordination, fait référence à un processus collectif. Le dictionnaire de l'académie française définit la coopération comme l'action de *concourir à une œuvre ou à une action commune*. Là où la coordination valorise les savoir-faire et les connaissances des "coordonnés" selon une gestion directive ou persuasive, la coopération valorise la production de groupe selon une gestion participative. Ainsi la coopération laisserait plus de place à l'innovation {Isoré, 2014}.

**Définition: Coopération**

La coopération consiste en la mise en relation d'entités qui participent volontairement à l'accomplissement d'objectifs communs. Elle nécessite la compatibilité des objectifs individuels ou leur accommodation consentie (les entités étant supposées de bonne foi).

Les collectifs cyber-physiques sont développés en héritant des systèmes embarqués en réseau : la prise de décision est cependant décentralisée. Ils sont donc constitués de nombreuses unités d'exécution autonomes qui coopèrent pour réaliser des tâches de contrôle, de communication, de traitement de données ou d'acquisition d'information. Elles sont en relation parce qu'elles communiquent par le biais de réseaux sans fil ou qu'elles interagissent via l'environnement physique. La tâche globale du système exploite cette coopération des différentes unités.

**Définition: Collectif cyber-physique (CCP)**

Un collectif cyber-physique est un système embarqué en réseau dans lequel les nœuds ont une autonomie de décision et coopèrent spontanément que ce soit pour participer à l'accomplissement d'objectifs du système global ou pallier des manques de connaissances ou de compétences individuels. Ces objectifs portent notamment sur l'état de leur environnement physique.

Construire des CCP impose de relever des défis complémentaires à ceux des systèmes embarqués hérités de la conception des SER :

**Défi 6. Complexité (*Complexity*)** Les logiciels des SER peuvent devenir des applications très sophistiquées impliquant plusieurs autres systèmes hétérogènes. Construire, maintenir et exploiter un modèle global est alors difficile. Ce défi milite clairement pour les approches décentralisées (et donc pour les CCP) dans lesquelles les sous-systèmes raisonnent à partir de vues partielles de l'environnement du système global et des autres sous-systèmes.

Il y a par exemple, dans un avion A380 {Salzwedel, 2011}, plus de 5000 dispositifs de contrôle électrique en réseaux et plus de 50 calculateurs qui travaillent en parallèle. Dans le contexte des technologies dites *fly by-wire* une architecture décentralisée a été développée pour réduire la charge de l'organe de contrôle central.

**Défi 7. Concurrence (*Concurrency*)** Les systèmes embarqués interagissent avec un grand nombre de processus physiques, contrôlent plusieurs actionneurs, interagissent entre eux et avec des humains. Les logiciels embarqués doivent réagir simultanément à plusieurs stimuli, provenant de leurs propres capteurs ou communiqués par d'autres systèmes embarqués, car ils correspondent à des évènements du monde physique. Ils seront donc généralement multi-tâches. La concurrence est un défi majeur dans la conception des SER et donc des glsCCP.
Par exemple, pour assurer la redondance d'un système de détection, les données (éventuellement contradictoires) provenant de plusieurs capteurs activés simultanément doivent être fusionnées.

**Défi 8. Hétérogénéité multiple (*Heterogeneity aggregation*)** L'hétérogénéité est une caractéristique intrinsèque des systèmes embarqués qui sont par définition des systèmes composés de logiciels et de matériels. Un autre type d'hétérogénéité réside dans les techniques de contrôle du temps (continues/discrètes), les actions immédiates et les traitements de plus longue échelle, la gestion synchrone ou asynchrone des événements etc. Les SER doivent assurer l'interopérabilité des nœuds tout en garantissant leur bon comportement global. En raison des différents types d'hétérogénéité, la recherche d'un modèle global est souvent difficile voire impossible. Il est alors nécessaire de combiner plusieurs modèles pour répondre au mieux à un problème donné. Le défi pour le concepteur consiste à caractériser les propriétés de l'agrégat d'hétérogénéité.
Un système de domotique dans lequel plusieurs sous-systèmes cohabitent (éclairage, chauffage, télévisions connectées, chaîne hifi connectée...) doit par exemple être capable d'appréhender globalement le confort d'un utilisateur.

**Défi 9. Intégration d'interfaces (*Interfaces integration*)** Les systèmes embarqués doivent intégrer ou composer simultanément des systèmes et des services différents. Ces exigences d'interopérabilité imposent aux logiciels (et aux matériels) embarqués de fournir des interfaces qui rendent possible la coordination des différents composants. Ces mécanismes de coordination peuvent mettre en œuvre des interactions sophistiquées alors même que les techniques de communication classiquement utilisées sont extrêmement simples. Les modèles d'interaction de la programmation classique comme les appels de procédure à distance (Remote Procedure Call (RPC)) ou même ceux de la Programmation Orientée Objet (POO) sont rigides. Le défi pour le logiciel embarqué est ici de bénéficier d'une technologie de composants qui accroit la flexibilité des définitions d'interfaces.
Des briques d'interopérabilité doivent par exemple être utilisées si un utilisateur veut permettre l'interopérabilité d'une équipe de drones et d'un réseau de capteurs précédemment déployé au sol.

**Défi 10. Reconfiguration et auto-organisation (*Reconfiguration/self-organization*)** Les caractéristiques précédentes contraignent les systèmes embarqués à modifier fréquemment leur comportement. Cette adaptation dépend des interactions et des actions qu'ils effectuent pour répondre aux attentes du système global. La gestion de la distribution est un défi parce que, plutôt que d'utiliser des tâches indépendantes très complexes sur des dispositifs

puissants, les CCP utilisent de nombreux petits sous-systèmes communicants fonctionnant comme un collectif. Le logiciel embarqué doit donc être "agile", s'auto-organiser et s'adapter aux ressources disponibles {Culler et al., 2001}.
A titre d'exemple, dans le contexte du contrôle de production flexible, les cellules d'usinage doivent être capable de se reconfigurer pour adapter le processus à l'introduction d'un nouvel outil.

**Défi 11. Mobilité (*Mobility*)** Dans le cas des systèmes embarqués mobiles, la localisation d'un nœud impacte la pertinence de son implication dans une coordination. Ces nœuds mobiles se coordonnent généralement en utilisant des communications sans fil qui sont généralement peu fiables et intermittentes. Les performances d'une coordination varient alors considérablement en fonction du temps et l'emplacement des nœuds {Bouroche and Cahill, 2008}. Le logiciel embarqué doit gérer la mobilité des nœuds : le défi est alors de permettre l'utilisation de mécanismes de coordination sensibles au contexte et compatibles avec des communications non fiables.
Cette propriété sera par exemple très importante pour une coalition de drones qui explore une zone sinistrée pour rechercher et secourir des victimes.

**Défi 12. Intégrité (*Integrity*)** Le maintien de l'intégrité des systèmes embarqués en réseau consiste essentiellement à assurer l'intégrité fonctionnelle du système global à l'aide des mécanismes traditionnels de tolérance aux pannes. Des défaillances peuvent survenir lorsqu'un partenaire est lui même défaillant ou à cause des problèmes de communication. Le défi réside ici dans la gestion de la confiance qu'ont chacun des nœuds avec leurs partenaires et aux mécanismes à implanter pour résister aux attaques externes {Ganeriwal et al., 2008}. Par exemple, pour prendre les bonnes décisions, un nœud qui utilise les mesures d'un autre devrait connaître la précision de cette mesure et la confiance qu'il peut avoir en sa véracité. Cette dernière valeur dépend elle-même de la confiance qu'il peut avoir dans les nœuds qui ont participé à l'acheminement de la mesure.
Le problème de la confidentialité des données est par exemple crucial dans le cas des applications militaires ou pour les dispositifs de téléopérations chirurgicales.

## 2.3 Pertinence d'une modélisation multi-agent

Les CCP peuvent être considérés comme des systèmes complexes artificiels impliquant des nœuds autonomes qui gèrent la décision, l'énergie, les services, les échanges de données, etc.

Nous avons toujours soutenu que les modèles issus des SMA embarqués sont particulièrement appropriés pour construire ces systèmes à condition qu'ils respectent les contraintes que nous avons préalablement exprimées sous forme de défis.

**Définition: Système multi-agent embarqué**

Un système multi-agent embarqué est un SMA dont la dimension physique impacte notamment la nature des agents, des interactions et de l'environnement. Un tel système multi-agent répond aux exigences des systèmes embarqués en réseau.

Les SMA proposent des approches, des modèles et des outils qui permettent de simplifier la conception des systèmes complexes (Défi 6). Ils sont traditionnellement utilisés pour la simulation de phénomènes sociaux, la résolution collective de problèmes, l'intégration de systèmes hétérogènes ou la commande décentralisée de systèmes. Ces trois derniers points sont particulièrement intéressants dans le contexte des systèmes embarqués.

Les avantages d'une approche multi-agent pour modéliser les CCP sont :

— au niveau de la conception :
  - une simplification du processus (de conception du système) car le faible couplage des agents permet de mieux appréhender la complexité globale (Défi 6) de l'application à construire,
  - une accommodation aux faibles ressources et une prise en compte des contraintes sous-jacentes aux défis 6 et 8 de par la non nécessité de donner aux agents l'accès à une vue complète du système global.

— pendant l'exécution du système :
  - un gain en robustesse car le système sera doté de capacités à faire émerger des comportements ou des structures organisationnelles adaptées à des situations non prévues lors de la phase de conception (Défi 10). Le système devient donc moins sensible aux changements de l'environnement (Défis 1 et 12),
  - une meilleure adaptation à la mobilité (Défi 11) de par le découplage fonctionnel inhérent des agents.

— au niveau de l'exploitation :
  - la représentation externe des interactions et l'organisation offrent de multiples possibilités à un observateur externe (Défi 9).

Depuis les années deux mille, les SMA semblent suffisamment matures pour leur utilisation dans un contexte industriel {Van Dyke Parunak, 2000}. Cependant, Munroe et al. {2006} souligne qu'il n'y a qu'un petit sous-ensemble des secteurs industriels dans lesquels ils sont utilisés. Quinze ans après cette première publication, ce sont toujours les mêmes secteurs qui sont affectés {Leitao et al., 2013; Müller and Fischer, 2014}.

Il est difficile de proposer une classification de ces applications, cependant nous allons citer quelques travaux dans des domaines particulièrement actifs en soulignant les défis majeurs qu'ils relèvent.

— Robotique collective. Il s'agit du domaine d'application des systèmes embarqués dans lesquels les SMA sont le plus naturellement utilisés {Huang et al., 2001}. Les agents représentent la partie décisionnelle des robots. Ils assurent la coordination des équipements physiques et la mise en œuvre des stratégies collectives. On retrouve de tels SMA dans la robotique mobile {Le et al., 2009; Takimoto et al., 2007; Jamont and Occello, 2013b} ou la gestion des processus industriels de fabrication {Monostori et al., 2006}. La mobilité (Défi 11) et la sécurité (Défi 5) sont des principaux aspects primordiaux de ce champ d'application.

— Réseaux sans fil ad-hoc. Les réseaux ad-hoc sont des réseaux dans lesquels la fonctionnalité

de routage des messages est répartie sur l'ensemble des nœuds du réseau. Ce domaine d'application a connu avec succès l'utilisation des SMA notamment grâce à ses différents modèles d'auto-organisation (Défi 10). Les SMA contribuent désormais à améliorer l'intégration de ces réseaux dans d'autres systèmes (Défi 9). Cela implique notamment d'adresser le problème de l'hétérogénéité (Défi 8) (capacités de fusion des données, de filtrage, d'accès aux données, etc.) {Hla et al., 2010; Jamont et al., 2010}.

— Transport et logistique. La gestion du trafic et la planification dans le cadre de la logistique sont des domaines très actifs des SMA mais la plupart des travaux s'arrêtent à la simulation. Les progrès actuels en robotique mobile et dans les réseaux ad-hoc permettent de dessiner le prochain déploiement de grandes applications {Chen and Cheng, 2010} de SMA embarqués.

— Domotique. La gestion du confort et de la sécurité des bâtiments a été longtemps un problème d'automaticiens. Le nombre croissant d'appareils et les interactions qui les accompagnent apportent d'importantes limites théoriques aux modèles actuels. La complexité des seuls modèles physiques impose de plus en plus aux automaticiens d'utiliser des approches décentralisées {Jamont et al., 2011}. La domotique d'aujourd'hui nécessite une gestion de l'hétérogénéité (Défis 8 et 9) et la gestion de l'énergie (Défi 4) est cruciale en raison des réglementations légales. L'auto-organisation (Challenge(10)) est nécessaire afin de réduire la complexité du système pour que les utilisateurs, qui doivent pouvoir ajouter ou retirer des équipements, ne soient pas impliqués dans de quelconques procédures de configuration.

— Surveillance automatisée. Les systèmes de surveillance automatisée sont à la fois des systèmes réactifs (il réagissent à la conjonction de différents événements détectés dans différentes sources vidéos) et pro-actifs (ils prennent par exemple automatiquement la décision d'alerter les humains pour lever le doute sur un événement). La complexité des traitements, les contraintes temps réel et la nature distribuée des réseaux de surveillance encouragent l'utilisation de SMA {Carrasco et al., 2010} notamment pour l'exploitation de décisions décentralisées (Défi 7), pour la gestion de l'intégrité fonctionnelle (Défi 12) et pour l'agrégation de données (Défi 9).

— Applications Radio Frequency Identification (RFID). Les agents ont récemment été utilisés pour développer des applications RFID d'identification, de localisation ou de traçabilité {Massawe et al., 2009}. Ce qui nous semble le plus intéressant dans ce champ d'application est la possibilité d'attacher des comportements déportés à des objets physiques dénués de capacités de calcul. Les agents fournissent des abstractions à ces objets physiques et produisent des comportements réactifs (Défi 1). Il permettent ainsi la création de sociétés virtuelles associées à des environnements physiques {Ricci et al., 2014; Jamont et al., 2014a}.

— Systèmes autogérés. Les SMA embarqués peuvent être utilisés pour l'autogestion de processus informatiques distribués (Défi (10)) permettant leur adaptation à des conditions non prévues (Défi (1)). Ce domaine applicatif est généralement traité par l'informatique autonome (autonomic computing {Kephart and Chess, 2003}) qui dote le système d'une capacité d'autogestion en orchestrant l'ensemble connu des d'éléments autonomes {Hassan et al., 2009}.
Visant l'intelligence collective du système, dans un SMA les entités ne sont pas uniquement vues comme des composants autonomes mais comme des entités sociales. Les agents sont

capables de communiquer mais aussi d'adapter dynamiquement leurs modes d'interactions et leurs comportements. C'est pour cette raison qu'ils sont d'ailleurs une source d'inspiration de l'informatique autonome {Tesauro et al., 2004; Huebscher and McCann, 2008}.

Si les SMA sont adaptés au monde industriel comme le montrait Van Dyke Parunak {2000}, les travaux de recherches menés au niveau des SMA embarqués n'ont toutefois pas nécessairement donné lieu à des déploiements dans le monde réel. La plupart des applications industrielles nécessitent un large éventail de fonctionnalités logicielles comme la planification, l'ordonnancement, la simulation ou l'aide à la décision. Le rapprochement des applications et du monde réel entraîne lui-même un rapprochement des fonctionnalités logicielles vers le matériel (diagnostic, commande et supervision de dispositifs/processus physiques, etc.). Un obstacle important au plus large déploiement industriel des "technologies" multi-agents est justement cette intégration matérielle des agents {Pechoucek and Marík, 2008}. Le tableau 2.1 illustre l'importance cruciale des travaux en cours sur ces SMA.

| Domaine | Intégration matérielle |
|---|---|
| Contrôle des processus industriels de fabrication | +++ |
| Logistique | + |
| Planification de production | ++ |
| Simulation | + |
| Contrôle de drones | + |
| Exploration d'espaces | ++ |
| Formation | - |
| Automobile | ++ |
| Chaînes d'approvisionnement | - |

(-) : L'intégration matérielle n'est pas un problème majeur
(+,++,+++) : L'intégration matérielle est (une exigence, une exigence importante, une exigence essentielle).

TABLE 2.1 – Importance de l'intégration matérielle dans les SMA industriels

## 2.4 Conclusion

Les SER sont constitués de nombreuses unités d'exécution reliées qui utilisent des technologies sans fil pour échanger des messages afin par exemple de coordonner leurs tâches de contrôle, de traiter ou d'acquérir des données. Nous nous intéressons aux SER dans lesquels les nœuds coopèrent pour répondre aux finalités que le système global aspire à atteindre. Nous considérons que les unités d'exécution qui les composent ont une autonomie de décision. Nous appelons ces systèmes des *collectifs cyber-physiques* (CCP) pour les dissocier des *systèmes cyber-physiques* (SCP) dans lesquels la décision n'est pas forcément décentralisée et qui ne proposent donc pas nécessairement des produits collectifs (services, structures...).

De nombreuses exigences, qui sont autant de défis à relever, doivent être prises en compte (tableau 2.2) lors de la construction des CCP.

| Défi # | Libellé |
|---|---|
| 1 | Réactivité (*Reactivity*) |
| 2 | Terminaison (*Liveness*) |
| 3 | Ponctualité (*Timeliness*) |
| 4 | Autonomie d'énergie (*Power Autonomy*) |
| 5 | Gestion de la sécurité (*Safety management*) |
| 6 | Complexité (*Complexity*) |
| 7 | Concurrence (*Concurrency*) |
| 8 | Hétérogénéité multiple (*Heterogeneity aggregation*) |
| 9 | Intégration d'interfaces (*Interfaces integration*) |
| 10 | Reconfiguration et auto-organisation (*Reconfiguration/self-organization*) |
| 11 | Mobilité (*Mobility*) |
| 12 | Intégrité (*Integrity*) |

TABLE 2.2 – Récapitulatif des défis liés à la conception des CCP

Les approches et les modèles issus des SMA peuvent jouer un rôle important dans la vie de ces systèmes si on leur donne la capacité d'être étroitement liés aux aspects matériels des applications ciblées.

# Chapitre 3

# Une approche multi-agent pour la conception des collectifs cyber-physiques

Dans le chapitre précédent, nous avons identifié les défis à relever pour construire les CCP. Nous avons vu que certains d'entre eux sont naturellement adressés par les SMA (notamment les défis de la complexité (Défi 6) et de la reconfiguration (Défi 10)).

Dans son étude des techniques d'intelligence artificielle applicables aux systèmes embarqués, Elmenreich {2003} restreignait l'utilisation des SMA aux systèmes qui n'intégraient pas de contraintes temporelles dures. Il sous-entendait que de nombreuses améliorations devaient être apportées aux SMA pour les rendre applicables aux systèmes les plus critiques. L'objectif de ce chapitre est justement de faire le point sur des améliorations possibles des approches multi-agents. A chacune des sections correspond une question méthodologique qui a motivé notre travail, et à laquelle nous répondons en formulant le problème, en rendant compte de la réponse apportée par les SMA et en proposant une solution possible que nous illustrons à partir d'activités spécifiques de la méthode Decentralized Iterative Approach for Multiagent Open Networks Design (DIAMOND).

## 3.1 Cycle de vie du développement

Les cycles de vie organisent les différentes tâches qui interviennent dans la création d'un système. Ils permettent entre autre de maîtriser le risque dans son développement.

**Définition: Cycle de vie**

Nous appelons cycle de vie un ensemble ordonné dans le temps d'*étapes* du développement d'un système de sa naissance à son retrait.

Une étape est un ensemble de *processus* organisés dans le temps et regroupés thématiquement (un objectif de haut niveau d'abstraction est visé). Le processus (ou phase) est aussi un ensemble

d'*activités* ordonnées dans le temps mais dont l'objectif est plus précis. Une *activité* est une action primitive entreprise dans le développement d'un produit logiciel.

**Problème.** La conception de systèmes embarqués commence traditionnellement par le recueil, puis l'analyse, des besoins formulés par le demandeur de l'application (figure 3.1). Il s'en suit une étape de partitionnement logiciel/matériel.

**Définition: Partitionnement logiciel/matériel**

Le partitionnement logiciel/matériel est l'étape du cycle de vie d'un CCP durant laquelle le concepteur décide des parties d'un système qui seront implantées sous forme "logicielle" (architecture à base de jeu d'instruction) et celles qui deviendront "matérielles" (architecture à base de blocs logiques matériels).

Un cahier des charges de la partie matérielle et un autre de la partie logicielle sont alors produits. Une implantation logicielle est généralement choisie à la fois pour la flexibilité qu'elle offre et pour simplifier la mise en œuvre des algorithmes. L'implantation matérielle est privilégiée essentiellement pour les performances qu'elle offre. Les parties matérielles et logicielles sont alors développées simultanément et sont intégrées en fin du processus de développement. De cette intégration, et des tests qui suivent, peut découler une remise en question de la conception ou de l'analyse de la partie logicielle, de la partie matérielle ou des deux. Plus loin, c'est le partitionnement lui-même qui peut être remis en question. Le risque qu'entraînent de mauvais choix rend l'étape de partitionnement cruciale dans cycle de vie d'un système embarqué.

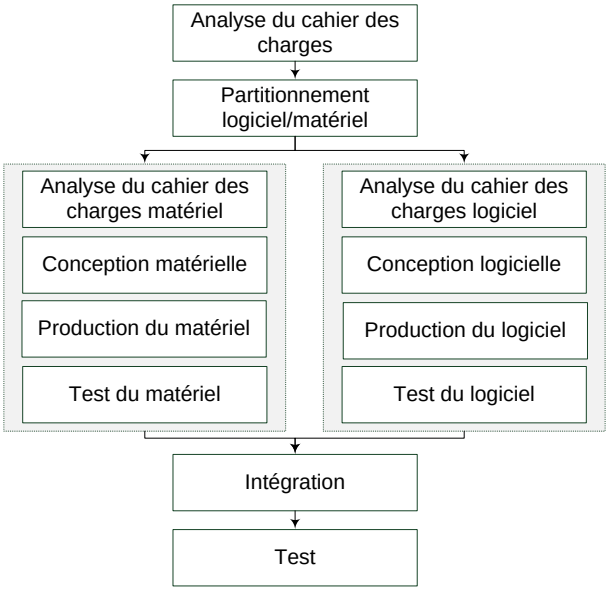



FIGURE 3.1 – Approche traditionnelle de la construction d'un CCP

Ainsi, plus de deux tiers des projets d'informatique embarquée n'atteignent pas 30% des performances attendues durant la phase de pré-conception Skazinski {2003}; Martin et al. {2007}. Les raisons sont essentiellement liées aux changements des spécifications en cours de projet ou à leurs

inadéquations avec les attentes du demandeur [1].

Ces risques sont accrus lorsque les systèmes conçus sont des collectifs notamment à cause des problèmes liés à la complexité (Défi 6) des applications adressées. Les cycles de vie traditionnels ne permettent pas une prise en compte de changements tardifs des spécifications. En outre, la criticité des applications nécessite de trouver les meilleurs compromis matériel/logiciel que ce soit pour diminuer un temps de réponse (Défis 1 et 3), la consommation d'énergie (Défi 4) ou le risque de défaillance (Défi 5). L'exploration des différents compromis nécessite de revenir aux étapes de conception précédentes (raffinement). Il faut aussi que le processus de conception permette la généricité (la caractéristique incrémentale des cycles de vie milite clairement pour la généricité). Enfin, il faut garder une trace de tous les paramètres qui ont motivé les différentes solutions retenues pour maîtriser le risque lié aux différents compromis.

**Travaux existants.** La plupart des méthodes multi-agents ne distinguent habituellement que deux grandes phases : l'analyse et la conception (par exemple MASE {DeLoach et al., 2001} ou Gaia {Wooldridge et al., 2000}). Très peu de méthodes traitent des autres phases et notamment celle de déploiement à savoir l'intégration des agents dans des environnements opérationnels. Les applications classiquement traitées par les SMA font que lorsque cette dernière étape existe, son objectif est le placement des agents logiciels sur des plateformes différentes d'exécution (comme dans MASSIVE {Lind, 2001}). Nous pouvons également citer le travail de Braubach et al. {2005} dont la phase de déploiement comprend la spécification de l'architecture physique du système et précise comment les logiciels doivent être déployés sur cette dernière à l'aide d'un outil de déploiement appelé *Agent Society Configuration Manager and Launcher (ASCML).*

En ce qui concerne les cycles de vie des méthodes multi-agents, la nécessité de processus itératif et incrémental est bien identifiée {Cernuzzi et al., 2005}. Les méthodes qui accordent une importance au déploiement essayent de positionner la phase de production du code en fin de processus pour des raisons de souplesse. Certaines d'entre elles proposent même des cycles de vie sur mesure afin de prendre en compte dynamiquement les modifications tardives d'exigences. C'est le cas de la méthode Agile-PASSI {Chella et al., 2006}.

**Nos propositions.** Les approches multi-agents se focalisent sur les aspects logiciels en omettant les aspects matériels qui ne sont généralement pris en compte qu'au niveau du déploiement (comme dans {Cossentino et al., 2003}). Les processus relatifs à la conception du matériel (figure 3.1, embranchement de gauche) sont remplacés par une activité qui consiste simplement à choisir une plateforme de déploiement pour les agents. La couverture du cycle de vie d'un système mixte logiciel/matériel par les SMA n'est donc que partielle.

On peut en conséquence affirmer que les approches multi-agents s'inscrivent dans le prolongement du développement traditionnel des logiciels embarqués (respectant le cycle de vie donné en figure 3.1). L'agent embarqué est alors vu comme la conjonction d'un agent logiciel et de sa plateforme d'exécution.

1. Voir par exemple l'enquête parue dans {Venture Development Corp., 2003} basée sur un sondage réalisé auprès de 400 entreprises travaillant dans le domaine des systèmes embarqués.

Cependant il a été constaté qu'il était nécessaire que le matériel et le logiciel soient conçus ensemble pour que la conception de systèmes mixtes logiciel/matériel soit à la fois fonctionnelle et optimale [2]. Ainsi, pour pallier les problèmes liés aux cycles de vie traditionnels du développement des systèmes embarqués, des méthodes alternatives de co-conception logicielle/matérielle (ou de codesign) sont apparues {Teich, 2012}.

**Définition: Méthode de codesign**

Une méthode de codesign logiciel/matériel est une méthode unique de conception, couvrant l'intégralité du cycle de vie d'un système embarqué en unifiant la conception des parties logicielles et matérielles. Elle permet l'abstraction des spécifications du logiciel et du matériel repoussant ainsi l'étape de partitionnement en fin du cycle de vie (figure 3.2).

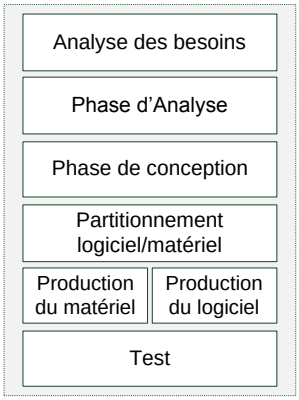



Figure 3.2 – Approche codesign de la construction d'un CCP utilisée dans DIAMOND

Afin de maîtriser au mieux le risque dans le développement des CCP (et donc des SMA embarqués), nous adoptons une telle approche. Elle permet notamment de mieux explorer l'espace de conception (les différents compromis logiciels/matériels) que les approches traditionnelles, afin de répondre à la fois aux critères de performance et aux exigences fonctionnelles.

L'exploration des différents cycles de vie, au vu des critères que nous avons exprimés lors de la formulation du problème, nous a amenés à choisir un cycle de vie en spirale {Boehm, 1988}. Le cycle de vie de la méthode DIAMOND est partagé en cinq phases (figure 3.3) :

**Phase A :** **Définition des besoins** qui permet de définir ce que le demandeur désire, d'identifier et structurer le fonctionnement global du système,

**Phase B :** **Analyse multi-agent** qui permet de décomposer le problème en une solution multi-agent,

**Phase C :** **Qualité** qui renforce la maîtrise du risque en posant la question des tests,

**Phase D :** **Conception générique** qui permet de construire le SMA sans distinguer le matériel et le logiciel,

2. Selon des critères aussi divers que l'encombrement du code, la quantité de mémoire nécessaire, le respect de contraintes temps réel, d'occupation de surface, de coût unitaire, de dégagement thermique etc.

**Phase E : Implantation** qui permet de partitionner le système et d'en faire la synthèse logicielle et matérielle.

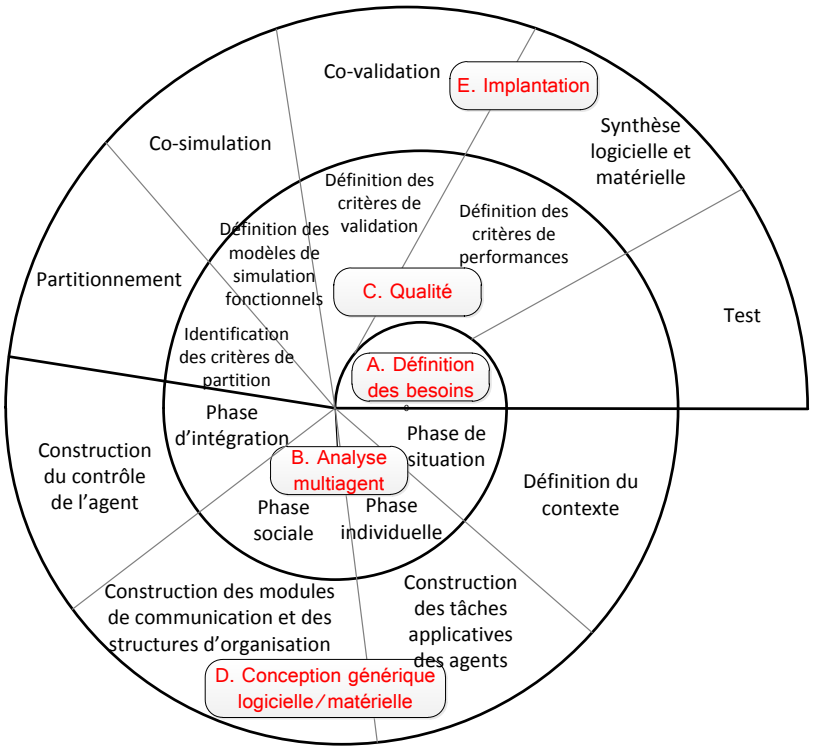



FIGURE 3.3 – Cycle de vie de la méthode DIAMOND

La phase de *Qualité* a été établie dans le cadre du projet Kurasu avec Rémy Thomas (Ingénieur Chercheur au CEA-LITEN Grenoble que nous avons encadré lorsqu'il préparait son Diplôme de Recherche Technologique (DRT)). Ces critères devaient être établis avant la phase de conception générique afin d'éviter que le concepteur, de par les choix de conception qu'il adopte en phase D, n'assouplisse certaines exigences. La phase de Qualité permet a minima de garder une traçabilité de ces relaxations.

Concernant la phase d'*Implantation*, nous avons collaboré avec l'équipe de Mouloud Koudil (LMCS à Alger) profitant de leurs travaux notamment dans le partitionnement logiciel/matériel. Les différents étudiants de DRT que nous avons encadrés avec Manuel Pezzin (chercheur au CEA-LETI) ont permis d'affiner le rôle de ces activités.

Plus globalement, cette démarche a bénéficié des retours d'expériences issus de mes propres travaux dans plusieurs projets industriels (FITT EnvSys avec la société CREATIME — instrumentation d'environnement naturels {Jamont et al., 2010}, FITT Palette avec la société EREIMA — encartage et palettisation d'étuis de médicaments) ou ceux de collègues de l'équipe (Projet SIET - Investissement d'Avenir - eSanté — assistance aux personnes fragiles {Mercier et al., 2013b}). Dans le cadre du projet SWANS (Projet MEDEA+ 2A204), Manuel Pezzin et Régis Guillermin l'ont utilisée pour concevoir un système de radiolocalisation permettant le suivi d'équipes d'intervention en

milieu à risque {Occello et al., 2008}. J'ai pu aussi échanger sur son utilisation lors du AOSE Technical Forum 2010 [3]. Enfin, Shadi Abras {2009} l'a appliquée à un problème de domotique durant sa thèse au Laboratoire d'Informatique de Grenoble (LIG).

## 3.2 Définition des exigences fonctionnelles et non fonctionnelles

**Problème.** Nous avons précédemment discuté d'une des spécificités majeures de l'utilisation des SMA pour modéliser des CCP : l'importance de la phase de déploiement. Les exigences en terme d'architecture physique doivent être exprimées au niveau des spécifications sous la forme de qualités attendues pour le système. Concevoir des SMA embarqués implique de traiter les Exigence Non Fonctionnelle (ENF) relatives à ce déploiement. Il est aussi nécessaire d'étudier les exigences en matière de sécurité (Défi 5). Nous entendons dans cette section par sécurité, la garantie que le système ne sera dangereux ni pour l'utilisateur humain, ni pour les biens matériels.

**Définition: Exigence fonctionnelle**

Exigence définissant une fonctionnalité du système à concevoir. Elle est en quelque sorte un service rendu par ce système.

**Définition: Exigence non fonctionnelle**

Exigence définissant une propriété du système à concevoir. Elle agit comme une contrainte sur les services rendus par ce système.

**Travaux existants.** La plupart des méthodes multi-agents se concentrent seulement sur les spécifications fonctionnelles alors que les ENF sont cruciales dans notre contexte. Certaines des méthodes multi-agents phares tentent d'intégrer ces techniques dans leurs dernières extensions.

- Harmon et al. {2009} proposent de mettre en évidence les conflits entre les différentes exigences fonctionnelles et non fonctionnelles recueillies dans O-Mase {DeLoach and García-Ojeda, 2010} afin d'aider le concepteur à établir une hiérarchie entre ces critères ;
- Blanes et al. {2009} avec RE-Gaia, étendent la méthode Gaia {Zambonelli et al., 2003} afin d'intégrer une approche de modélisation des exigences en s'appuyant sur la structure organisationnelle du SMA ;
- Tropos {Bresciani et al., 2004} permet de modéliser les systèmes par raffinements progressifs dans lesquels les ENF sont vus comme des *softgoals* (un objectif sans critère précis) ;
- Werneck et al. {2007} montrent qu'ADELFE {Bernon et al., 2003; Picard and Gleizes, 2004} prend en compte quelques exigences non fonctionnelles dans son activité de définition des besoins consensuels (activité 3). Ces exigences sont, par exemple, des pré-requis pour le stockage de volumes de données importants, les capacités d'interaction homme-machine, la disponibilité continue du système et son adaptation. Certaines contraintes d'exécution (distribution, aspects multi-tâches) sont aussi prises en considération.

3. Voir `http://www.pa.icar.cnr.it/cossentino/AOSETF10/`

Un des principaux problèmes est d'exprimer les exigences de façon à simplifier leurs spécifications, à les rendre réutilisables et à les considérer tout au long du cycle de vie. Certains outils reposant sur des techniques basées sur les ontologies {Lindoso and Girardi, 2006} ou des approches dirigées par les modèles {Naji et al., 2004} ont été proposés.

La maîtrise de ces exigences non fonctionnelles se fonde sur la qualité de leur expression. Cependant, ces exigences peuvent évoluer avec la représentation et la compréhension du système par son concepteur : il est donc important d'assurer leur traçabilité {Castor et al., 2004; Thangarajah et al., 2011} ce qui permettra, par exemple, de relier la contrainte d'une fonctionnalité à l'acteur à l'origine de sa capture.

Hormis les contraintes de dimensionnement ou les contraintes temporelles, des exigences liées au contexte physique à étudier sont celles de la sécurité. Étant en relation très étroite avec les dispositifs physiques, les systèmes embarqués peuvent rendre dangereuses certaines actions sur les humains ou sur l'environnement. Leveson {2004} avance que les accidents dans les systèmes complexes se produisent en raison d'interactions mal comprises ou mal réalisées entre les humains et les machines. Dangers et risques doivent donc être pris en compte durant la conception et le fonctionnement de ces systèmes. Ce problème est pourtant rarement abordé. Quelques travaux cependant se penchent sur la sécurité mais en tant qu'inviolabilité du système {Bresciani et al., 2004}. Ces approches ne considèrent ni les applications physiquement déployées ni la sécurité des utilisateurs. Aussi, il est naturel de se demander comment ces spécifications doivent être considérées dans le domaine multi-agent. Certaines études sur les dangers dans les systèmes complexes {Alexander et al., 2008} défendent que l'analyse ne peut être réalisée qu'en évaluant l'exposition au risque, soit subi, soit causé par chaque entité du système. La méthode doit alors partir non pas uniquement d'un recueil des besoins des utilisateurs, mais aussi d'une modélisation des risques du système complet. Certains travaux ont tenté d'adapter les techniques classiques largement utilisées dans les industries manufacturières au champ multi-agent.

L'analyse des modes et effets des fautes (Failure mode and effects analysis (FMEA)) est une identification qualitative des dangers qui aide à détecter des modes de fautes potentielles basées sur l'expérience passée. L'objectif est d'étudier les effets de ces échecs et comment ils peuvent affecter l'utilisateur. Ebrahimipour et al. {2010} proposent une structure d'agent et l'utilisent pour assurer la gestion de la sécurité dans un SMA au moyen de diagrammes de diagnostic de fautes essayant de résoudre les limitations de la FMEA dans les systèmes complexes comprenant de nombreux composants. Influencés par les FMEA, {Rodriguez-Fernández and Gómez-Sanz, 2010} proposent un langage permettant d'exprimer des critères d'auto-gestion afin d'amener le système à s'auto-configurer et à s'auto-protéger.

Sterling and Taveter {2009} présentent la sécurité comme un critère de qualité pour un SMA. Les auteurs proposent d'améliorer les méthodologies multi-agents en utilisant des études de "danger et opérabilité" (HAZard and OPerability study (HAZOP)). L'approche HAZOP est une procédure systématique pour déterminer les causes et les conséquences des déviations d'un comportement normal. Plusieurs applications industrielles utilisant des SMA ont mis en œuvre des règles HAZOP {R. Lakner an Nemeth et al., 2006; Johnson, 2005}.

Les contraintes de sécurité établies comme des exigences doivent être plus loin traduites en

modèles d'analyse. Dans Tropos, Bresciani et al. {2004} proposent de les prendre en compte comme des contraintes d'interaction à travers la spécification de cas d'utilisation. Un risque peut alors être considéré comme l'occurrence de conséquences négatives non-souhaitées d'un événement et intégré en tant que tel dans le comportement des agents. {Raja et al., 2009} introduisent la notion de conception conservative : il s'agit de la capacité d'un agent à évaluer seul son comportement global à partir de ses actions et interactions. Asnar and Giorgini. {2007} traitent la sécurité comme un contexte pour Tropos étendant les buts pour prendre en compte les risques. Une phase de simulation peut permettre d'évaluer les politiques de sécurité en menant une étude basée sur les risques du modèle du système tout entier {Despotou et al., 2009}.

**Nos propositions.** La première étape de la méthode DIAMOND (figure 3.4) permet la définition des besoins. Elle commence par une phase d'*approche informelle du problème (A.1)* qui inclut la définition du contexte du système. Dans un second temps, on se préoccupe de capturer les besoins fonctionnels des utilisateurs : *étude des acteurs (A.2)* et *étude des cas d'utilisation (A.3)*. Un acteur représente un ensemble de rôles cohérents que les utilisateurs d'un système jouent quand ils sont amenés à interagir avec lui. Un cas d'utilisation est une fonctionnalité proposée par le système. On procède alors à l'*étude des besoins en services (A.4)* de chacun des acteurs. Les diagrammes de séquence permettent de concevoir des scénarii d'utilisation du système et de représenter simplement les échanges de messages entre le système et les acteurs.

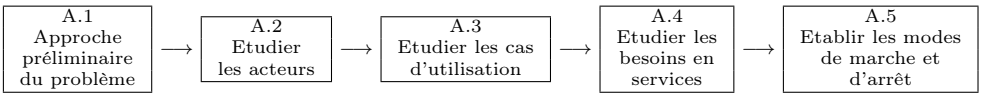



FIGURE 3.4 – L'étape de définition des besoins de la méthode DIAMOND

L'*étude des modes de marche et d'arrêt (A.5)* constitue une réponse originale pour structurer le fonctionnement global du SMA embarqué en prenant en compte des ENF. En effet, si généralement on souhaite que le système fonctionne en autonomie pour répondre aux besoins fonctionnels, il est nécessaire de connaître précisément d'autres comportements :

- Comment doit-être le système avant l'arrêt ? Quand le SMA va subir un arrêt prolongé, il est nécessaire de porter une attention particulière à l'état de repos des différents composants physiques du système.
- Dans quel état doit-être un agent physique pour être étalonné/réglé ? Quel est l'impact de cette tâche sur le SMA ? Les agents équipés de capteurs et d'effecteurs doivent en effet être étalonnés pour par exemple pallier les problèmes de dérive des mesures ou l'usure des pièces mécaniques qui composent les effecteurs électromécaniques.
- Comment prendre en compte les arrêts d'urgence ? La présence d'hypothétiques acteurs humains dans l'environnement des agents nécessite la possibilité d'arrêts d'urgence en vue d'assurer la sécurité. Quand les équipements sont mobiles et avec une source d'énergie embarquée, le problème n'est pas aussi simple qu'avec des systèmes centralisés.

Même si le problème est par la suite résolu de manière décentralisée, cette organisation des modes de fonctionnement au niveau global présente l'avantage d'être facilement interprétée par le deman-

deur et l'utilisateur. De plus, bien qu'à intelligence décentralisée, le système doit se soumettre à une réglementation qui impose la présence d'arrêts d'urgence pour les systèmes physiques pouvant nuire à l'intégrité d'une personne ou d'un bien. Cette activité participe à la spécification des contraintes au niveau global. L'articulation au niveau local se fera par l'intégration dans le comportement des agents de ces contraintes.

Cette activité est fortement inspirée du Guide d'Etude des Modes de Marche et d'Arrêt (GEMMA) {Adams and Paques, 1988; Moreno and Peulot, 2009} longtemps utilisé par les automaticiens des systèmes à événements discrets. Il permet de structurer le fonctionnement d'un système automatisé. D'un point de vue opérationnel, cela consiste à organiser des GRAphes Fonctionnels de Commande des Étapes et Transitions [4]. Nous avons spécialisé son fonctionnement pour répondre à nos besoins dans le contexte de la conception de SMA embarqués.

La figure 3.5 présente l'outil fourni au concepteur pour l'aider, au niveau global, à identifier différents modes de fonctionnement du SMA et donc de capturer les ENF. C'est en quelque sorte une *checklist* qui permet de vérifier notamment que le concepteur du CCP n'a pas oublié de mode particulier. Afin que le SMA ait une disponibilité maximale, il convient de prendre en compte le problème de la sécurité à tous les niveaux de la conception du SMA. L'outil permet de faire le point et de réfléchir plus en avant sur le problème de la sécurité.

De manière analogue à l'utilisation du GEMMA dans le contexte des automates à événements discrets, chaque rectangle d'état est caractérisé par :

— le symbole de dénomination qui traduit l'appartenance à une famille de modes : (Si,Ri,Fi) respectivement pour (Stop, Run, Failure),

— la désignation du mode ou de la procédure : par exemple "Test du SMA en séquence" est associé au rectangle d'état identifié par le symbole $R5$.

Un rectangle d'état correspond en quelque sorte à un contexte du SMA. Le concepteur va donc, dans un premier temps, sélectionner les rectangles d'état pertinents pour le SMA étudié. Il spécifie les conditions des liens qui permettent la transition entre les différents contextes.

Cette activité s'intéresse à trois différentes familles de modes (figure 3.5) que sont les *modes de marche*, les *modes d'arrêt* et les *modes de défaillance*.

Les *modes de marche* (table 3.1) définissent les états pendant lesquels le SMA fonctionne en autonomie en vue de répondre (ou se mettre en état de répondre) aux exigences fonctionnelles. On observe donc des états d'étalonnage ou de réglage du système. Ils spécifient notamment :

— les démarrages du SMA à froid : le système était arrêté, il faut donc initialiser les composants logiciels et matériels du SMA (agents, objets de l'environnement),

— les démarrages du SMA à chaud : certains composants du SMA avaient perdu leurs états suite à une panne ou à une erreur qui avait entraîné l'arrêt total ou partiel du SMA (problèmes d'alimentation électrique, perturbations électromagnétiques, etc.).

Les *modes d'arrêt* (table 3.2) vont permettre d'identifier les états d'arrêt. Le SMA ne peut fonctionner indéfiniment. On distinguera notamment comme traditionnellement dans les systèmes embarqués :

4. Dérivés des réseaux de Petri, les GRAFCETs sont des outils permettant la représentation graphique de systèmes automatisés séquentiels à événements discrets.

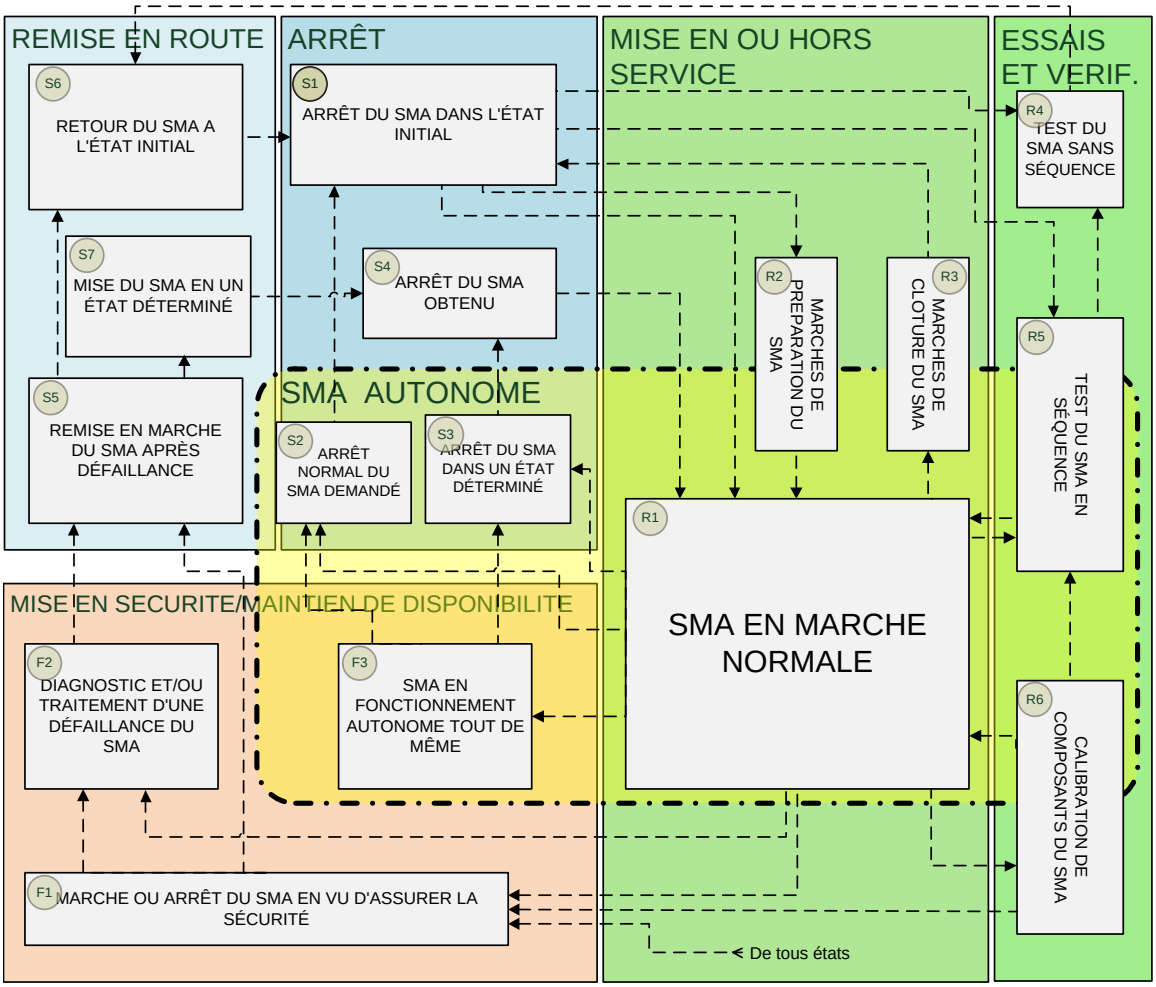



FIGURE 3.5 – Grille d'analyse fournie pour le recueil des exigences non fonctionnelles

| État | Description |
|---|---|
| R1 | *SMA EN MARCHE NORMALE* Dans cet état, le SMA est dans son mode de fonctionnement normal, c'est-à-dire qu'il fonctionne en autonomie : les agents sont autonomes. |
| R2 | *MARCHES DE PRÉPARATION DU SMA* Cet état concerne les procédures de mise en œuvre des opérations (manuelles ou non) qui doivent être effectuées afin d'être dans les conditions rendant possible le fonctionnement autonome du SMA (sa marche normale). |
| R3 | *MARCHE DE CLÔTURE DU SMA* Cet état contient les procédures qui vont assurer la transition de la marche normale vers un état d'arrêt prolongé du SMA. |
| R4 | *TEST DU SMA SANS SÉQUENCE* Cet état spécifie les opérations qui vont permettre de tester les différents capteurs et actionneurs des objets de l'environnement ou des agents. L'agent ainsi testé perd son autonomie : sa boucle de décision est désactivée et il est commandé manuellement par exemple par l'opérateur de maintenance. |
| R5 | *TEST DU SMA EN SÉQUENCE* Cet état permet l'exécution de procédures de test plus complexes. Des séquences d'actions, pouvant mettre en œuvre des capacités de coordination et de coopération des agents, vont être évaluées par un opérateur. |
| R6 | *CALIBRATION DE COMPOSANTS DU SMA* Ce mode permet la calibration, l'étalonnage et l'ajustement des capteurs et des effecteurs des agents ou d'objets de l'environnement. |

TABLE 3.1 – Modes de marche — Procédures de fonctionnement

— les clôtures : il s'agit d'arrêts pour causes "normales". Par exemple, la session de travail du SMA est terminée ou ses tâches doivent être provisoirement interrompues par décision d'un opérateur ayant autorité ;

— les arrêts en vue d'assurer la sécurité : ils sont déclenchés par des opérateurs humains via un bouton coup de poing (arrêt d'urgence), par des systèmes matériels (une barrière à ultrasons qui détecte l'entrée d'un opérateur humain dans une zone réputée dangereuse) ou des capteurs virtuels (des processus qui mesure le temps de réaction d'un opérateur suspect...).

Les arrêts de sécurité sont bien entendus toujours prioritaires et font l'objet de nombreuses normes européennes (EN292, EN418 etc.) et extra-européennes (ISO 10218, ISO 11161, ISO 13855, etc.).

| État | Description |
|---|---|
| S1 | *ARRÊT DU SMA DANS L'ÉTAT INITIAL* Dans cet état, les agents sont alimentés en énergie, mais leur comportement autonome n'est pas encore activé. |
| S2 | *ARRÊT NORMAL DU SMA DEMANDÉ* Dans cet état, on demande au SMA de s'arrêter mais en acceptant qu'il poursuive temporairement son fonctionnement autonome pour se mettre dans un état qui ne présente aucun risque ni pour son intégrité ni pour la sécurité des opérateurs humains. Cet état est utilisé par exemple lorsqu'on veut changer la source d'alimentation d'un agent. |
| S3 | *ARRÊT DU SMA DANS UN ÉTAT DÉTERMINÉ* Le SMA n'arrêtera sa marche normale que lorsqu'un état spécifique sera atteint. Cet état permet de conduire le SMA dans un état de reconnaissance. |
| S4 | *ARRÊT DU SMA OBTENU* Cet état permet de conduire le SMA dans un arrêt différent de celui obtenu en S2. |
| S5 | *REMISE EN MARCHE DU SMA APRÈS DÉFAILLANCE* Après une défaillance, il permet de ramener le SMA dans un état qui l'autorisera à procéder à une remise en marche normale sans passer par l'état initial. |
| S6 | *RETOUR DU SMA Á L'ÉTAT INITIAL* Dans cet état, certains composants du SMA peuvent être contrôlés manuellement pour le mettre dans un état spécifique. |
| S7 | *MISE DU SMA EN UN ÉTAT DÉTERMINÉ* Dans cet état, le SMA va revenir dans un état précédent de manière manuelle ou automatique avant de revenir à son fonctionnement autonome. |

Table 3.2 – Modes d'arrêt

Les *modes de défaillance* (table 3.3) vont permettre de spécifier les procédures de sécurité permettant de réagir à une défaillance ou un incident. Lorsqu'une défaillance intervient, il faut le signaler (alarmes sonores, affichage lumineux...) et donner des consignes aux composants du SMA etc. Bien entendu, toutes les défaillances ne sont pas prévisibles : l'adaptation du SMA permettra d'en pallier certaines et, dans les cas critiques, les opérateurs déclencheront les procédures d'arrêts d'urgence.

| État | Description |
|---|---|
| F1 | *MARCHE OU ARRÊT DU SMA EN VU D'ASSURER LA SÉCURITÉ* Cet état concerne les séquences d'actions spéciales qui doivent être déclenchées dans une situation d'urgence. Cet état inclut non seulement l'arrêt mais aussi l'exécution de procédures pour limiter les effets de la défaillance. |
| F2 | *DIAGNOSTIC ET/OU TRAITEMENT D'UNE DÉFAILLANCE DU SMA* Cet état permet, lorsqu'une défaillance est apparue, d'examiner le SMA. L'opérateur de maintenance pourra alors diagnostiquer l'origine d'une défaillance. L'application d'une procédure appropriée permettra le redémarrage du système. |
| F3 | *SMA EN FONCTIONNEMENT AUTONOME TOUT DE MÊME* Dans certaines circonstances, il est nécessaire de laisser le SMA agir en autonomie mais en lui appliquant des restrictions. Certains composants du SMA peuvent être stoppés voire remplacés par des humains. |

Table 3.3 – Modes de défaillance

Le cycle de vie d'exécution du SMA peut être caractérisé en identifiant certaines boucles dans le guide d'étude (figure 3.6). Ces boucles caractéristiques sont :

— Le *cycle de fonctionnement normal* du SMA correspond à la boucle ($S1 \rightarrow R1 \rightarrow S2 \rightarrow S1$). Le SMA est à l'arrêt, un opérateur ayant autorité demande au système de fonctionner de manière autonome pour répondre aux exigences fonctionnelles. Ensuite, lorsque la session de travail est terminée, l'opérateur informe le SMA qu'il doit s'arrêter. Le SMA se remet à l'état initial qui correspond à son état de repos.

— La *marche de réglage* correspond à la boucle $S1 \rightarrow R5 \rightarrow R4 \rightarrow S6 \rightarrow S1$. Le SMA quitte l'état $S1$ (Arrêt du SMA dans l'état initial) et passe en $R5$ (Test du SMA en séquence). Ainsi, l'opérateur peut vérifier et ajuster le bon fonctionnement de certaines tâches des agents. Il peut agir plus finement sur le réglage des primitives d'action ou de perception dans le mode $R4$. Une fois les vérifications effectuées, le système passe à l'état $S6$ (Retour du SMA à l'état initial). Dans cet état, le système va atteindre les conditions initiales qui lui permettront de transiter par la suite vers son fonctionnement autonome.

— La boucle d'*arrêt de sécurité* correspond à l'enchaînement $R1 \rightarrow F1 \rightarrow F2 \rightarrow S5 \rightarrow S6 \rightarrow S1 \rightarrow R1$. Elle intervient lorsque le SMA rencontre une défaillance.

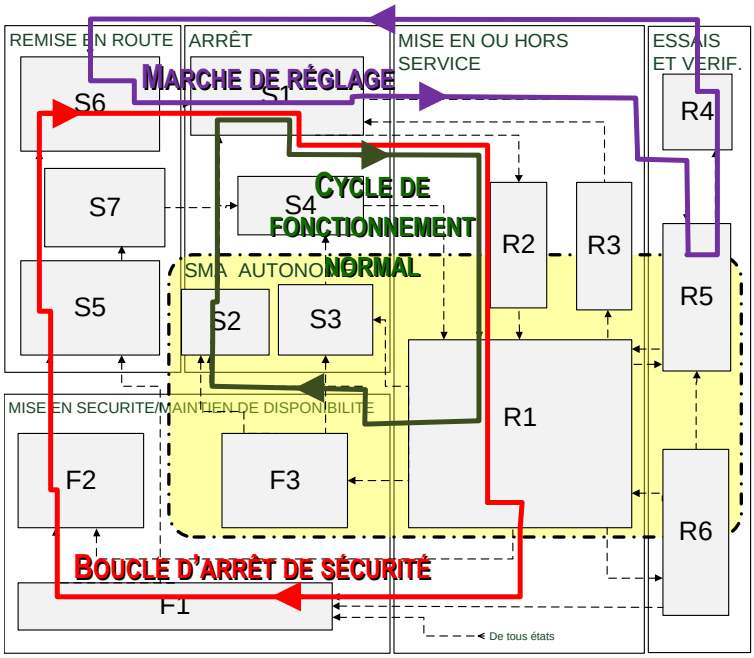



FIGURE 3.6 – Boucles caractéristiques d'un fonctionnement particulier du SMA

Nous avons illustré cette activité en nous inspirant d'un cas d'étude que nous avions déjà travaillé dans le cadre du groupe de travail Colline (GDR I³ , Groupe de travail Colline [5] et pour lequel existe l'outil de simulation *MultiAgent for Supply Chaining (MASC)* [6]). L'exemple complet est publié dans {Jamont and Occello, 2013a}.

5. `http://www.irit.fr/COLLINE/index.html` — Collectif, Interaction, Émergence
6. Développé au sein de l'équipe SMAC à l'IRIT par André Machonin

## 3.3 Particularités d'analyse

**Problème.** Nous avons vu que certains aspects du système global doivent être capturés et spécifiés sous la forme d'exigences (cas de la sécurité que nous spécifions sous la forme de contraintes non fonctionnelles). D'autres aspects, eux aussi directement liés à la spécificité des SMA embarqués, doivent être pris en compte lors de la phase d'analyse par l'intermédiaire de modèles.

**Travaux existants.** Considérons les travaux existant selon les défis que nous avons listés.
*Réactivité et Ponctualité (Défis 1 et 3) :* Ces défis font partie des points qui nous semblent les plus durs à traiter. Elmenreich {2003} fait valoir que les capacités temps réel ne peuvent pas être garanties de manière fiable à cause du faible couplage des agents (qui sont généralement asynchrones). En effet, il implique un non déterminisme temporel et la possibilité d'interblocages. Il exclut la possibilité d'utiliser les SMA dans le cas de systèmes embarqués fortement contraints par le temps. Cependant, très tôt, Occello and Demazeau {1996} ont évalué l'impact des aspects temps réel sur la conception des agents et montré qu'ils doivent être pris en compte pour chacune des capacités des agents et à chacun des niveaux de la conception comme rappelé sur la figure 3.7.

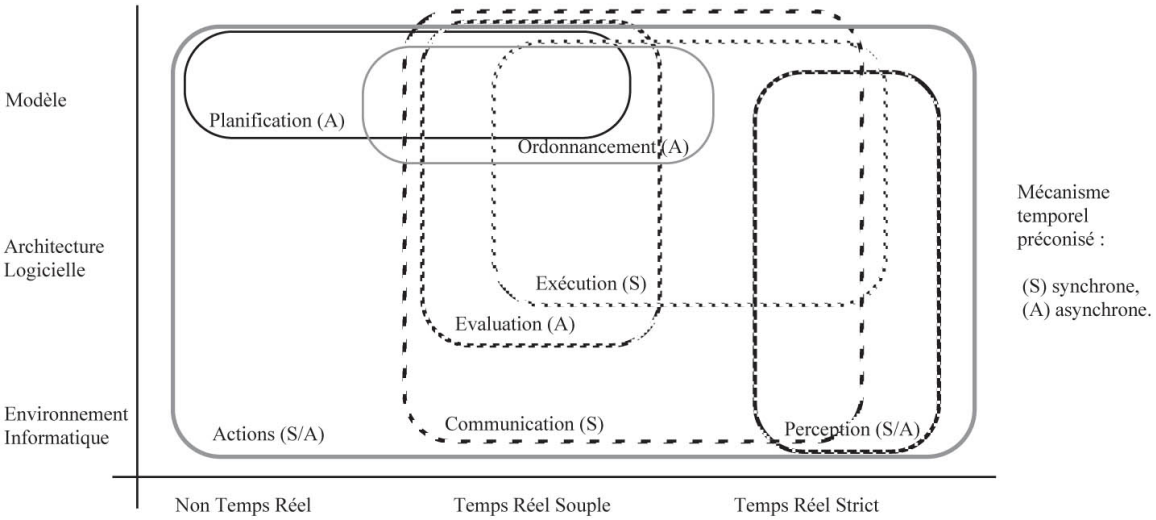



FIGURE 3.7 – Distribution de la prise en compte des contraintes, figure tirée de {Occello and Demazeau, 1996}

Plusieurs travaux des SMA traitent du temps réel en adoptant des modèles formels spécifiques. Par exemple Hutzler et al. {2005} représentent le comportement d'agents réactifs par des automates temporisés afin de vérifier que le système global se conforme à des impératifs de correction logique (le système fait ce qu'il doit) et de correction temporelle (le système se conforme à un ensemble de contraintes temporelles). Julian and Botti {2004} proposent une extension de la méthode MESSAGE pour construire des SMA temps réel qu'ils nomment avec RT-MESSAGE. Ils prennent en compte des contraintes temporelles dans les comportements des agents et leurs modèles d'interaction à partir d'une ontologie des objectifs et des tâches qu'ils enrichissent avec des spécifications temporelles.
*Gestion de l'énergie (Défi 4) :* Durant cette dernière décennie, la problématique de la gestion de l'énergie est devenue source de nombreux travaux dans la communauté des SMA. Ce défi est souvent

traité via des structures organisationnelles ou des comportements spécifiques d'agents. Pour gérer la consommation d'énergie des nœuds d'un réseau sans fil, Ambuhl et al. {2004} proposent d'utiliser des agents égoïstes et d'implanter des mécanismes de récompense (problème d'optimisation multi-agent). Takimoto et al. {2007} et Shakshuki and Malik {2007} présentent une organisation hiérarchique pour diminuer l'énergie dépensée par les communications respectivement dans le cadre de la robotique collective et des réseaux de capteur.

*Gestion de la sécurité (Défi 5) :* Nous traitons ce défi comme s'il s'agissait d'une exigence non fonctionnelle et donc dans la phase de définition des besoins. D'autres travaux de la communauté l'adressent dans la phase d'analyse. Par exemple avec Tropos, Bresciani et al. {2004} la considèrent comme un problème d'interaction qu'ils représentent via un modèle de cas d'utilisation d'acteurs. Le risque peut alors être considéré comme une conséquence négative, et donc indésirable, d'un événement. Sa détection permet à l'agent d'adopter un comportement particulier. Raja et al. {2009} introduit la notion de conception conservatrice (conservative design) qui est la capacité d'un agent à évaluer son comportement global à partir de ses actions et interactions. Cette évaluation peut permettre la détection d'une situation de non sécurité. Asnar and Giorgini. {2007} gèrent la sécurité en étendant les buts des agents pour prendre en compte les risques qu'ils identifient.

*Hétérogénéité multiple (Défi 8) :* Elmenreich {2003} souligne qu'elle est le plus souvent abordée à travers l'interaction. Les SMA embarqués doivent faire face à différentes représentations de données et à différentes sémantiques.

*Gestion de la mobilité (Défi 11) :* Dans les SMA, la plupart des travaux qui prennent en compte cette problématique s'intéresse à maintenir la connectivité entre des agents communicants via des réseaux sans fil. En particulier, les contributions multi-agents de la robotique collective essaient de résoudre ce problème en subordonnant la décision des agents au modèle d'organisation {Bouroche and Cahill, 2008; Le et al., 2009}.

*Intégrité (Défi 12) :* Concernant ce défi, Ganeriwal et al. {2008} relèvent que les SMA embarqués peuvent avantageusement tirer profit de la notion de réputation et de confiance. Ces derniers proposent des mécanismes pour que les informations qui transitent par des réseaux sans fil (en particulier les réseaux de capteurs) soient protégées des attaques malicieuses.

En ce qui concerne les formalismes utilisés dans la phase d'analyse, les méthodes multi-agents utilisent généralement des notations et des modèles d'une seule origine {Bernon et al., 2002} comme UML (MASE, AAII, MESSAGE, PASSI). D'autres méthodes utilisent plusieurs notations comme Tropos {Castor et al., 2004} qui utilise i* pour modéliser les connaissances et AUML pour modéliser les interactions et les plans. DESIRE utilise des notations basées sur les graphes pour modéliser les connaissances et des notations spécifiques pour décrire les tâches.

**Nos propositions.** L'analyse multi-agent est le cœur de la méthode DIAMOND. Cette analyse travaille à deux niveaux différents (figure 3.8). Au niveau société, le système est vu comme un tout qui veut répondre à des besoins fonctionnels en respectant des exigences non fonctionnelles. Au niveau individuel, le concepteur focalise son attention sur les agents du système. Ils coopèrent notamment pour trouver dans le collectif les connaissances et les compétences qui leur font défaut.

Vu le caractère mixte logiciel/matériel, il nous faut dans cette démarche, étudier une spéciali-

sation de la définition des différents axes de la décomposition Agent, Environnement, Interaction, Organisation (AEIO) proposée par Demazeau {1995} pour les étendre à la description du "matériel". Il est à noter qu'à ce niveau de la méthode DIAMOND, aucune frontière logicielle/matérielle fixe n'existe pour ce qui concerne la spécification du contrôle.

Pour couvrir les différentes phases d'un cycle de vie, nous pensons comme Herlea et al. {1999} que plusieurs formalismes sont nécessaires à différents niveaux d'abstraction. DIAMOND ne propose pas de formalisme dédié mais réutilise plusieurs formalismes existants et langages spécifiques (Finite State Machine (FSM), Hardware Description Langage (HDL) ou des notations plus répandues dans la communauté multi-agent (notations Unified Modelling Langage (UML), Agent Unified Modelling Langage (AUML) etc.).

L'étape d'analyse s'appuie sur les travaux antérieurs de l'équipe Occello M. {2001}. Elle regroupe quatre phases (figure 3.8). Les itérations sur ces phases permettront de produire une décomposition multi-agent du problème. Les différentes facettes AEIO du SMA seront finalement intégrées dans les agents.

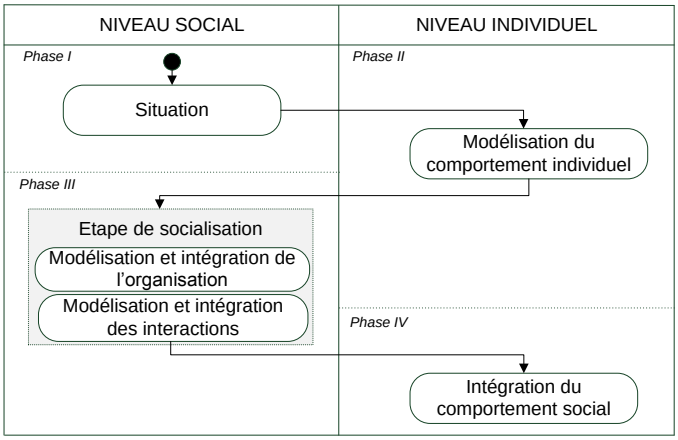



FIGURE 3.8 – Étape de l'analyse multi-agent

La *phase de situation (B.1)* consiste à définir les limites de l'application, à caractériser les agents et l'environnement. On commencera par *caractériser l'environnement* afin d'identifier et recueillir les informations qui portent principalement sur le contexte de conception. Ces caractéristiques permettent d'identifier les paramètres qui interviennent dans la représentation du monde qu'ont les agents et conditionneront l'architecture de l'agent. Il faudra ensuite *déterminer les entités actives et passives*. Une entité active possède de l'autonomie vis-à-vis de son environnement et des autres. Elle peut régir des règles d'interactions entre les différentes entités. Généralement, les entités passives sont des ressources du système qui répondent quasiment uniformément aux sollicitations des autres entités. Dans notre contexte, une entité peut contenir du logiciel (active : un dispatcheur de services, passive : un fichier) et du matériel (active : un robot, passive : une balle) mais peut aussi se présenter plus simplement sous la forme d'un jeu de contraintes qui module des interactions. A partir des interactions utilisateurs/système identifiées via les diagrammes de séquence, on va caractériser les entités qui participent aux interactions entités/utilisateurs. Enfin, il faudra *déterminer les entités*

*que l'on modélisera par des agents.* Pour cela, on s'appuie essentiellement sur les descriptions des entités actives faites lors de l'activité précédente. Les entités passives ne seront étudiées que par la perception qu'en auront les entités actives.

La *phase individuelle (B.2)* permet de s'intéresser aux aspects externes et internes des agents en ayant pour objectif l'élaboration d'un comportement purement individuel. L'aspect interne consiste à définir ce qui est propre à l'agent, c'est-à-dire ce qu'il sait faire (primitives d'actions ou services) et ce qu'il connaît (sa représentation des A - E). L'aspect externe concerne la définition des médias mettant en rapport l'agent avec le monde extérieur, c'est-à-dire ce qu'il peut percevoir (représentation du monde) et comment il le perçoit (diagramme de contexte). L'activité va mettre en exergue, pour chacun des agents, les besoins en capteurs et en effecteurs. Il faudra donc produire un tableau des perceptions/actions qui permettra d'identifier la nature de ces perceptions, les capteurs nécessaires pour accomplir cette perception, et les effecteurs afin d'éventuellement modifier l'état de ce qui est perçu.

Dans l'activité A.3 (Étude des cas d'utilisation), nous avons établi une vue fonctionnelle du système : il nous faut désormais voir le même problème d'un point de vue plus matériel et centré sur l'agent. Les diagrammes de contexte sont un cas particulier de diagrammes de flots de données qui spécifient les interfaces physiques d'un système. Ils définissent les composants de l'environnement tels qu'ils sont perçus par le système, en les identifiant comme une série de terminaisons rattachées aux interfaces du système.

Nous allons ainsi, grâce à cette activité, définir le contexte de l'agent physique c'est-à-dire caractériser l'environnement physique dans lequel il évolue. Il est nécessaire de s'intéresser entre autres aux interactions possibles, même non désirées, entre le système et son environnement (actions et perceptions). Le caractère physique du système rend nécessaire de s'intéresser dans un premier temps à la nature des tâches. En effet, l'instanciation du système de raisonnement commence par une phase monoagent de construction de plans d'action. On va définir les missions confiées aux agents et des actions requises. Il est possible de définir des actions que l'agent n'est pas capable d'effectuer et qu'il délèguera au moment de l'exécution. Les actions travaillent, dans la plupart des cas, à partir des données disponibles dans la représentation de l'environnement détenue par l'agent. Il faut donc détailler cette représentation à partir des besoins exprimés lors de la spécification des actions. On commencera par s'intéresser à la représentation que l'agent a de lui-même (identifiant, niveau d'énergie etc.). Afin de garantir la réalité de ces données, la définition des capacités de perception utiles s'impose. Les perceptions nécessitées par ces tâches peuvent être définies d'une manière analogue aux actions : les *perceptions primitives* qui sont les perceptions non décomposables et les *perceptions composées* qui sont construites à partir d'autres perceptions c'est-à-dire des éléments de la représentation du monde. Les besoins en données de l'agent, qui ne sont pas satisfaits, devront être clairement définis et mis en avant. Lors de la phase suivante, il sera en effet nécessaire de savoir si les autres agents peuvent répondre à ces besoins. Dans le cas contraire, il faudra contacter le demandeur et réitérer le processus d'analyse. On établit alors, sans tenir compte des différentes solutions technologiques ou des contraintes de conception, le Diagramme de Contexte, élément spécifique original dans un cycle SMA, qui va nous permettre d'identifier le système (ou certaines fonctionnalités principales) par rapport aux flux d'entrées et de sorties. Dans l'activité précédente, on a défini les actions et

les perceptions de l'agent : à ce niveau on va les instancier en précisant les échanges d'informations entre les différents organes physiques qui exécuteront les actions/perceptions primitives. La remontée des contraintes temporelles et de performance se fait en terme de temps de réponse et de taux d'échantillonnage au niveau du diagramme de contexte.

La *phase sociale (B.3)* permet de focaliser sur les interactions entre agents et sur l'organisation afin d'élaborer un comportement purement collectif. A ce niveau du cycle de vie, nous nous concentrons sur les actions qui peuvent initialiser des interactions avec d'autres agents et répondre à tous les besoins en interaction. Dans un premier temps, il est nécessaire de préciser la perception des autres agents. On indique ce que chaque agent peut savoir à tout instant et comment il peut le savoir. La caractérisation des primitives de communication se fait ensuite selon quatre points. Tout d'abord on s'intéresse aux *médias de communication* qu'offre l'environnement physique. Un médium est le support de transmission permettant le passage des informations d'un équipement à un autre (fibre optique, cuivre, air etc.). On étudiera l'*adressage* qui doit être mis en place : conversations bipartites ou multipartites, diffusion à des groupes (multicast), inondation des autres agents (broadcast). La *persistance* sera aussi précisée afin de savoir si les messages doivent être disponibles pour une certaine durée. Enfin, on se penchera sur la *localité* des communications : identifier s'il est possible ou non que les envois de message imposent des sollicitations de leur entourage. Si le cahier des charges contient des contraintes techniques à ce sujet, il est important de les prendre en compte. Seule l'analyse des besoins en communication des agents est approfondie. A ce stade de l'analyse, on s'intéresse à la définition des services évolués intégrant des interactions avec les autres agents et aux éventuels contrats : on définit les protocoles d'interaction. On s'intéresse aussi aux perceptions que l'on peut obtenir grâce à d'autres agents.

Comme dans la décomposition des SMA traditionnels, on va étudier les possibilités de collaborations, la formation de coalitions, etc. On associera par exemple des schémas d'interactions aux rôles à l'aide d'une notation de type AUML.

Enfin, la *phase d'intégration (B.4)* va permettre la socialisation des individus (les agents) c'est-à-dire d'intégrer les influences sociales aux comportements individuels des agents par l'intermédiaire des capacités d'évaluation de la communication et de la perception détaillées dans le modèle de l'agent. Les influences possibles sur les comportements purement individuels doivent être analysées et intégrées dans les agents. Jusqu'ici la décomposition a occulté la notion de contrôle de l'agent, c'est-à-dire la façon dont il va gérer son attention, sa décision, et enchaîner ses actions. Cet aspect que l'on peut qualifier de mixte se greffe sur les deux précédents (social et individuel). On va donc prendre en compte les perturbations dûes aux autres agents (identifiées précédemment). C'est par l'intégration des influences sociales dans les agents que l'on va donner une dynamique au SMA.

Afin d'illustrer cette étape, nous allons évoquer un système de radiolocalisation que nous avons participé à développer avec le CEA-LETI qui permettait de suivre une équipe de secours dans un milieu à risque.

La figure 3.9 illustre l'utilisation d'un diagramme de contexte afin de caractériser, sans tenir compte des différentes solutions technologiques ou des contraintes de conception, l'agent par rapport à ses flux d'entrées d'information et de sorties. Le diagramme de contexte définit ainsi les limites de l'agent en fixant les interfaces avec l'environnement.

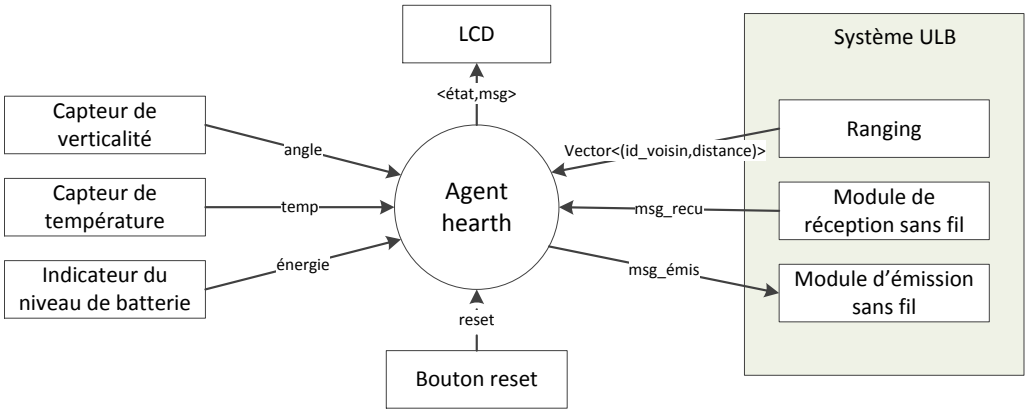



FIGURE 3.9 – Diagramme de contexte

La table 3.4 est un exemple d'outil que nous utilisons pour construire les règles de contrôle de l'agent. Il permet de décrire l'ensemble des situations de réaction, d'évaluation et de décision. Les colonnes figurant dans ce tableau sont :

- Observation : grandeurs et valeurs observées sur la représentation du monde (équivalent à un état donné du monde)
- Perception : messages et contenus reçus
- Paramètres de la décision : les grandeurs qui vont entrer dans la fonction de décision
- Plans activables : les actions (ou suite d'actions) susceptibles d'être déclenchées suite à la validation des conditions d'évaluation
- Urgence : priorité de la décision (immédiatement avec suspension des autres tâches (préemption), après la fin des tâches en cours, etc)
- Fonction de décision ou de coût : décide de la pertinence, des plans (actions) à activer et de l'urgence de cette activation en fonction de l'état du monde.

Il faut comprendre la première ligne du tableau ainsi : lorsqu'on démarre l'agent (cette requête vient de l'IHM) et sous réserve que son niveau d'énergie soit conforme avec son état (une règle implantée) l'agent va s'initialiser. Cette procédure va lui coûter de l'énergie.

Ce cas d'étude est détaillé dans les articles {Occello et al., 2008; Jamont et al., 2009}.

## 3.4 Conception générique

**Problème.** Les systèmes embarqués mettent en jeu des logiciels monolithiques et généralement dépendants des plateformes. Ils sont difficiles à maintenir, à mettre-à-jour et à réutiliser dans d'autres applications. Exporter un logiciel d'une plateforme spécifique à un autre est souvent un travail délicat.

Dans les méthodes de génie logiciel, la conception générique {Roques and Vallée, 2003} fait référence au développement d'une solution qui répond aux spécifications techniques mais qui reste entièrement indépendante des aspects fonctionnels. Dans notre cas, elle fait en plus référence à l'indépendance (et donc l'abstraction) de l'aspect matériel et logiciel.

| OBSERVATION | PERCEPTION | PARAMETRES | PLANS | URGENCE | COUT |
|---|---|---|---|---|---|
| | Démarrage (IHM) | Quota d'énergie vérifié | Initialisation de l'agent (base de données, modules) | Aucune | Energétique |
| | Agent opérationnel | Quota d'énergie vérifié | Recherche de réseau | Aucune | Energétique |
| | Réseau détecté | Quota de trafic et d'énergie vérifié | S'associer au réseau | Aucune | Energétique, trafic |
| | Pas de réseau | Quota de trafic et d'énergie vérifié | Démarrer un nouveau réseau | Aucune | Energétique, trafic |
| | Associé au réseau ou nouveau réseau démarré | Quota d'énergie vérifié | Se localiser (initialisation interruption périodique) | Aucune | Energétique, localisation |
| | Demande d'association d'un agent | Quota de trafic et d'énergie vérifié | Associer et répondre à l'agent | Aucune | Energétique, trafic |
| Se localiser (interruption périodique) | | Fréquence de localisation, Quota de trafic et d'énergie vérifié | Requête de position et/ou *ranging (+confiance)* | Aucune | Energétique, trafic |
| | Réponse de position et/ou *ranging (+confiance)* | Quota d'énergie vérifié | Estimer sa position et mettre à jour la base de connaissances | Aucune | Energétique |
| | Requête de position | Quota de trafic et d'énergie vérifié | Retourner sa position et l'indice de confiance | Répondre dans la seconde (application) | Energétique, trafic |
| Veille standard (interruption périodique) | | | Mise en veille standard | Aucune | Aucun |
| Grande inactivité | | Grande inactivité | Mise en veille prolongée | Aucune | Aucun |
| | Requête de mise à jour logicielle | Quota de trafic et d'énergie vérifié | Lancer procédure de mise à jour | Aucune | Énergétique, trafic |
| | Détection d'organisation défaillante | Quota de trafic et d'énergie vérifié | Lancer procédure d'élection, de négociation ou de nomination | Urgent | Énergétique, trafic |
| Trafic, énergie, confiance en la position, agent mobile | | Trafic, énergie, confiance en la position, agent mobile | Changer la fréquence de localisation et/ou la priorité des tâches | Aucune | Énergétique, trafic, localisation |

TABLE 3.4 – Ensemble des actions et des données associées

La plupart des travaux qui s'intéresse au développement de logiciels embarqués indépendants des plateformes matérielles utilisent des composants.

**DÉFINITION: Composant**

Unité indépendante de production et de déploiement, qui remplit une fonction spécifique et qui permet aux développeurs de définir des schémas logiques (code, description matérielle) réutilisables. Il est conçu de manière à fonctionner facilement avec d'autres composants pour créer des fonctions plus complexes ou des applications {Frenot, 2002}.

De nombreux travaux ont visé l'adaptation de modèles de composants pour les systèmes embarqués. Beaucoup de modèles et d'outils sont axés sur le logiciel comme par exemple PBO {Stewart et al., 1997}, Koala {van Ommering et al., 2000} , ReFlex {Bard, 2003} , PECOS {Jawawi et al., 2006} , PICML {Balasubramanian et al., 2007} et Rubus {Hanninen et al., 2008}. Les composants ont connu dans le monde des systèmes embarqués un important succès qui a amené le domaine à s'approprier le terme de "composant sur étagère" utilisé traditionnellement dans le monde du génie logiciel {Li et al., 2009} (Off-The-Shelf (OTS), Commercial OTS). Peu de travaux tentent cependant d'unifier composants logiciels et matériels {Bunse and Groß, 2006}. Des travaux aboutis comme {W.

Cesário et al., 2004} s'intéressent à des problèmes de très bas niveau propres au domaine du codesign (cosynthèse des parties logicielles et matérielles dans ce cas précis). {Feiler and Gluch, 2012} est un ouvrage consacré à *Architecture Analysis and Design Language* (AADL) qui propose de manipuler trois types de composants : les composants relatifs aux applications logicielles (thread, processus...), aux plateformes d'exécution (processeurs, mémoires, bus...) et aux compositions.

**Travaux existants.** Si tout n'est pas modélisable, ni plus encore implantable avec un même modèle de SMA, les mêmes schémas se répètent souvent à travers différentes applications. Il est nécessaire de proposer des outils basés sur la réutilisabilité. Il faut en effet permettre d'impliquer des modèles génériques développés pour chacune des notions (agent, environnement, interactions, organisation) en les instanciant pour les applications visées.

L'objectif de ce niveau est donc d'obtenir un système opérationnel à partir de la description informelle du SMAs. Les composants sont là aussi de très bons candidats pour construire des applications industrielles multi-agents comme le souligne Briot {2014}.

Combler le fossé entre l'analyse multi-agent et la conception est un problème important toujours d'actualité qui a monopolisé beaucoup d'énergie. L'utilisation de composants pour l'implantation des agents a préoccupé de nombreux chercheurs au début des années 2000 {Kendall and Malkoun., 1997; Guillemet et al., 1999; Meurisse and Briot., 2001; Occello et al., 2002; Vercouter, 2004} mais les travaux sur des cycles de développement complets de SMA intégrant une dimension componentielle sont rares comme le soulignait Lind {2001}; Brazier et al. {2002}. Plus récemment, se rapprochant du monde réel, des contributions visent à construire les agents en utilisant des composants dans le cadre de la commande de systèmes {Polaków and Metzger, 2009} où des composants Labwiew sont utilisés pour construire des systèmes multi-contrôleurs distribués.

Pour les CCP, une interrogation importante concerne la façon dont les modèles multi-agents peuvent être incorporés dans des systèmes logiciels/matériels reconfigurables (Défi 10). Certaines approches présentent des partitionnements verticaux pour constituer des sociétés hybrides d'agents matériels purs et d'agents logiciels purs {Naji et al., 2004}. D'autres travaux utilisent des partitionnements horizontaux : l'agent est vu comme la conjugaison d'une partie physique (le corps de l'agent) et de logiciels (l'esprit). Meng {2005} fait valoir que le modèle d'agent Belief, Desire, Intention (BDI) peut être considéré comme une structure unifiée permettant, à partir d'heuristiques, de partitionner les systèmes en agents logiciels et en agents matériels.

**Nos propositions.** Pour notre part, nous nous sommes intéressés à l'utilisation des composants comme unité opératoire pour construire les SMA mixtes logiciels/matériels essentiellement en raison du découpage fonctionnel clair inhérent à leur utilisation qui facilite l'étape de partitionnement. De plus, l'analogie composant logiciel/composant électronique permet aux outils du logiciel et de la description du matériel de partager une vision unifiée (voir le découpage entre description externe/interne dans les spécifications VHDL). Il est cependant vrai que la simplicité de compréhension et de mise en œuvre par "l'utilisateur" du composant qui peut se faire via une programmation graphique était intéressante et avait motivé l'outil préliminaire *MultiAgent System Codesign tool*

*(MASC)* développé durant ma thèse [7]. De plus la décomposition des fonctions participe à diminuer la complexité du système (Défi 6) et l'incitation à la généricité était attrayante.

La conception générique se déroule en 4 étapes (figure 3.10) :

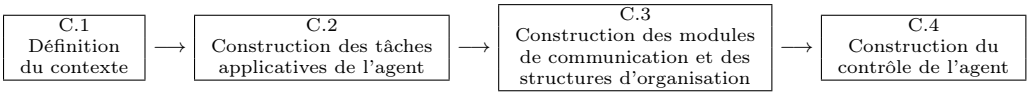



FIGURE 3.10 – La conception générique logicielle et matérielle dans DIAMOND

On commencera par *définir le contexte* (C.1.) qui a été esquissé durant l'étape d'analyse. Cette définition va permettre de *préciser le diagramme de contexte* en procédant à des choix technologiques pour les actionneurs et les effecteurs de chaque agent (exemple en table 3.5). Les paramètres qui influent sur ce choix sont traditionnellement les performances, le coût, la consommation électrique, le rendement, la fiabilité etc. En fonction de ces choix, il faut caractériser les interfaces qui permettront de numériser les informations. On complètera ainsi les informations apparaissant sur le diagramme de contexte. On devra alors *choisir une architecture pour les différents agents* en accord avec le modèle choisi et les contraintes spécifiées précédemment. Cette architecture servira de patron pour la conception détaillée en composants (on pourra l'exprimer suivant le formalisme RT-UML {Douglass, 1999}).

| Information | Description |
|---|---|
| Reset | Actif sur front montant |
| Verticalité | Angle de verticalité (valeur flottante codée IEEE 754, 32b) |
| Température | Température en celius (valeur flottante codée IEEE 754, 32b) |
| Énergie | Énergie restante exprimée en pourcentage (valeur flottante codée IEEE 754, 32b) |
| <état,msg> | Visualisation de l'état du nœud (8b) et de messages spécifiques d'alerte (chaine ASCII terminée par le caractère nul) |
| Vector<voisin_ id, distance> | Vecteur donnant la distance de chaque voisin (agent dans la portée du module ULB). (voisin_id : uint 16b, distance : en mètres, réel codé IEEE 754, 32b) |
| msg_recu | Séquence de bits ($emetteur_{uint16b}$, $destinataireuint16b, longueur\ donnees_{ushort8b}$, $donnees_{0-2048b}$) |
| msg_émis | Séquence de bits ($emetteur_{uint16b}$, $destinataireuint16b, longueur\ donnees_{ushort8b}$, $donnees_{0-2048b}$) |

TABLE 3.5 – Spécifications associées au diagramme de contexte SART

On pourra alors *construire les tâches applicatives de l'agent* (C.2.). A une tâche sera associé un composant (qui pourra lui-même être construit par l'assemblage de plusieurs composants) comme l'illustre la figure 3.11. On va *Construire la coquille externe de l'agent* c'est-à-dire les parties de l'agent qui vont acquérir l'information provenant des interfaces capteurs afin d'établir la représentation du monde (figure 3.11, composants $C_{i_x}$). On construira les parties de l'agent qui vont agir sur le monde extérieur pour changer son état (composants $C_{o_x}$). Chacune des interfaces entre les capteurs/effecteurs et le cœur de l'agent seront des composants. L'objectif de ces composants est de construire (échantillonner, coder) une information exploitable par la future partie décisionnelle de l'agent à partir de ce qui a été mesuré. On va alors *construire la coquille interne de l'agent* en

7. En raison des ressources demandées par le développement et la mise à jour d'un tel outil, MASC est resté au stade de démonstrateur.

s'attachant à la création des actions évoluées : actions paramétrées, composées et situées. Chacune de ces actions sera représentée par un composant qui sera connecté via des ports à un ou plusieurs composants liés à la coquille externe (composants $C_{om_x}$). Il en sera de même pour les différentes perceptions (composants $C_{im_x}$).

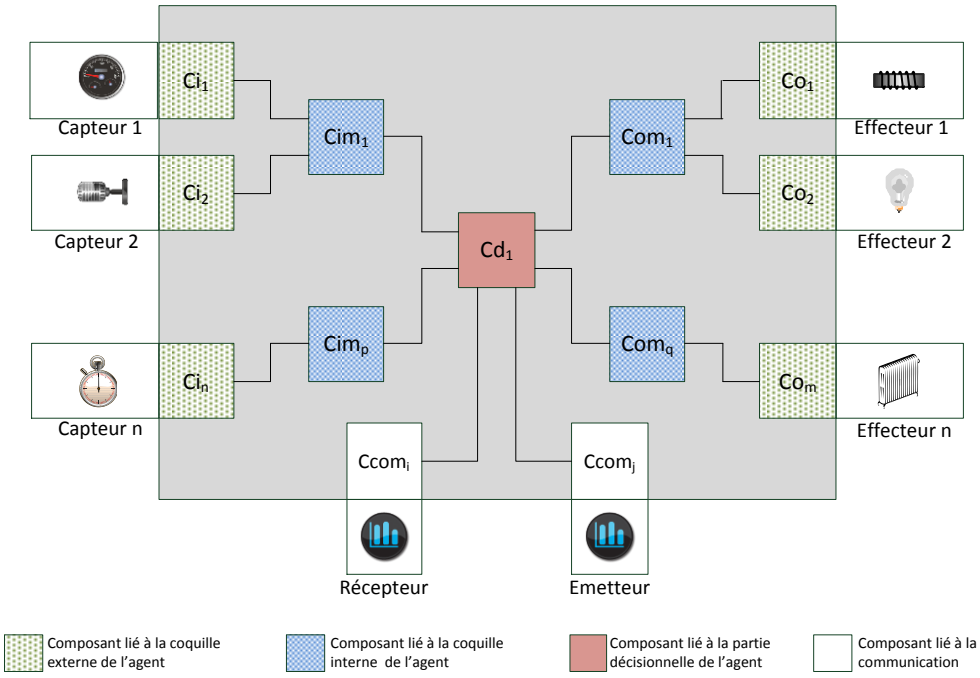



FIGURE 3.11 – Construction générique d'un agent avec des composants mixtes

On poursuivra la construction de l'agent en effectuant la *construction des modules de communication et des structures d'organisation* (C.3.). Cette construction se fait, comme précédemment, avec des composants (composants $C_{COM_i}$ et $C_{COM_o}$). Dans notre contexte, le protocole (spécifié en AUML) sera traduit en un automate à états finis. L'avantage des automates à états finis est qu'ils permettent facilement de générer du code logiciel ou des descriptions matérielles.

La dernière activité de cette étape sera la *construction du contrôle de l'agent (C.4)* Cette phase compose avec l'élaboration des comportements des composants/agents. Les parties "évaluation" et "décision" de l'agent seront créées dans cette étape. Quand un message est reçu, le composant/agent doit être capable de fournir une réponse appropriée, de choisir les comportements qui doivent être exécutés. Cette exécution de tâches va changer alors l'état du composant (et donc de l'agent).

Il est important de noter que nous exploitons ici le potentiel du cycle en spirale. La séquence C.1 à C.4 d'enrichissement des composants agents se fait par déclinaison des résultats des phases B.1 à B.4 de l'itération.

Nous avions développé l'outil MASC pour permettre cette conception générique puis l'implantation du système. Son développement a été le fruit du travail de plusieurs étudiants de licence professionnelle et d'un élève ingénieur. Cédric Reynier, Guillaume Cartier et Philippe Guillaud ont particulièrement contribué à l'avancement de l'outil. MASC a été présenté au groupe de travail ré-

gional *Systèmes Complexes et Multi-Agents* (SyCoMas) mais son développement a été rapidement arrêté en raison des importantes ressources que sa réalisation demandait et de son éloignement du champ thématique des SMA.

## 3.5 Implantation

**Problème.** Il existe différentes mises en œuvre possibles d'une solution multi-agent embarquée : l'implantation distribuée ou non distribuée.

*Implantation non distribuée :* Les solutions non distribuées sont appréciées par l'industrie car elles passent outre une des barrières psychologiques qu'occasionne la décentralisation, à savoir la délégation de décision à un ensemble de systèmes autonomes. En effet, dans ce cas, la partie décisionnelle du système est située dans un unique endroit clairement identifié (typiquement une armoire électrique) et l'activité de maintenance paraît plus facile.

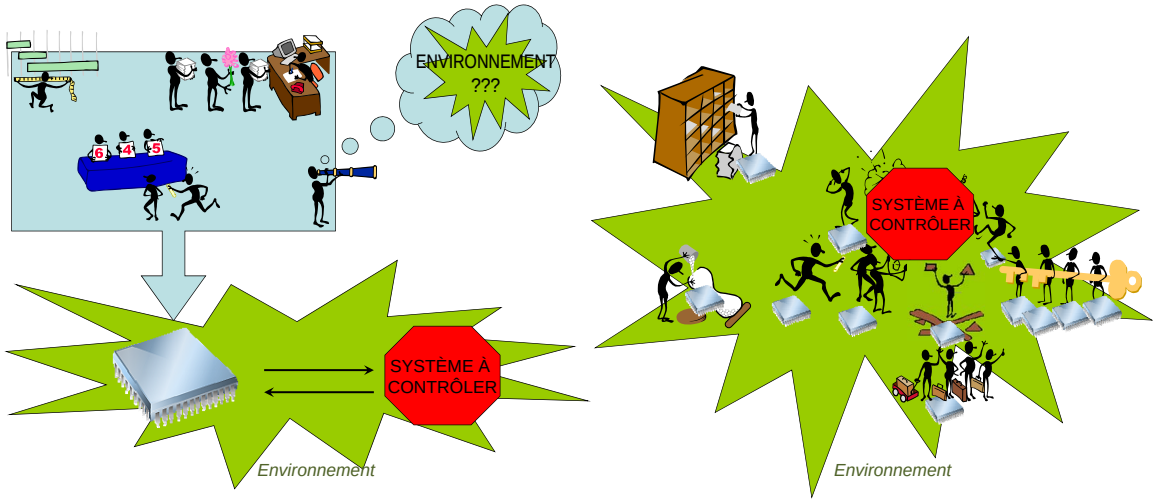
Dans de tels systèmes, le SMA embarqué commande les dispositifs physiques à la place des commandes traditionnelles venant de l'automatique. Dans ce type de cible, les agents du système multi-agent sont sur une seule et unique plateforme reliée à tous les capteurs et à tous les effecteurs qui composent le système global. Chaque agent raisonne à partir de sa propre représentation du monde réel. Les agents participent bien à une résolution collective du problème. Si le SMA est déployé sur un Multi-Processor System on Chip (MPSoC), on peut introduire le concept de SMA sur puce (MASoC, figure 3.12 a). Le SMA peut reposer sur un exécutif temps réel introduisant un pseudo-parallélisme comme le propose {Julian and Botti, 2004}. Cette architecture est typique des applications de gestion de chaîne logistique.

*Architecture distribuée :* En cas d'implantation distribuée, les agents sont déployés sur différentes unités qui composent le système global. Ils interagissent de manière dynamique pour résoudre le problème selon leur propre représentation partielle de l'environnement. Chaque agent peut être déployé comme un tout ou comme un ensemble de composants en utilisant architectures orientées services {Spanoudakis and Moraitis, 2007; Wu et al., 2007; Jaszczyk and Król, 2010}.

De tels SMA sont des systèmes multi-"agents embarqués" plutôt que des "systèmes multi-agents" embarqués comme dans le cas précédent. Si une plateforme matérielle est une puce qui n'abrite qu'un seul agent, le concept de l'agent sur puce (Agent on Chip (AoC), figure 3.12 b) peut être introduit. Les réseaux d'instrumentation sans fil dans lesquels les capteurs sont modélisés par des agents sont un exemple typique.

Quelle que soit la solution choisie, les agents sont des candidats très sérieux pour offrir de bonnes propriétés d'intégration (Défi (9)).

Permettre une implantation de l'agent plus proche du matériel est un bon moyen d'assurer une meilleure réactivité (Défi 1) que traditionnellement, mais conduit à poser le problème de l'affectation de comportements complexes à un objet physique qui n'a pas toujours les ressources suffisantes pour les exécuter. Si le matériel ne dispose pas suffisamment de ressources, on peut déporter le comportement sur un serveur distant. Ce serveur héberge un avatar de l'objet physique (voir figure 3.13). Par exemple, si on souhaite donner un comportement intelligent à un pot de yaourt taggé RFID, on ne peut pas intégrer le code dans l'objet physique.




(a) Implantation non distribuée - MASoC (b) Implantation distribuée - AoC

FIGURE 3.12 – Implantation distribuée/non distribuée d'une intelligence décentralisée

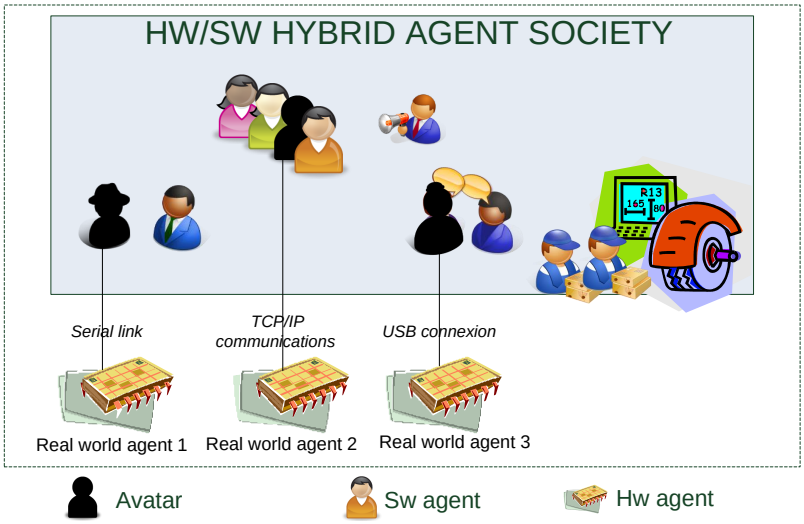



FIGURE 3.13 – Un agent du monde réel interagissant avec des agents logiciels via son avatar

**Travaux existants.** Plus les aspects décisionnels de l'agent sont implantés dans le matériel, plus il est difficile d'assurer les capacités de reconfiguration (Défi 10) en raison du problème des spécifications des descriptions matérielles {Meng, 2005}.

L'étape d'implantation est concernée par la prise en compte des exigences temporelles (Défi 3). A ce niveau, il est possible de faire cohabiter au sein d'un même système des agents matériels respectant des contraintes de temps dures avec des agents logiciels moins efficaces selon ce critère {Naji et al., 2004}. L'utilisation de comportements déportés oblige à prendre en compte les possibles connexions intermittentes et des délais d'acheminement des messages {Meng, 2005} .

**Nos propositions.** La principale utilisation du co-design figure dans le *partitionnement* logiciel/matériel qui sera fait des différents composants de chaque agent en fonction des spécifications technologiques éventuellement mises en exergue dans le recueil des besoins. Il nous est indispensable de nous interroger sur les critères qui vont orienter nos décisions vers leur implantation logicielle ou matérielle. Pour chacun des composants pour lesquels se pose la question du partitionnement, on va hiérarchiser les critères que l'on souhaite satisfaire. On en déduira le choix du partitionnement et on s'interrogera quant aux interfaces logicielles/matérielles à implanter. Nous présentons dans le tableau 3.6 les critères de partitionnement qui nous semblent pertinents pour les agents à partir de différents travaux effectués dans le co-design (notamment {Kumar, 2002}). Lorsque l'on fait collaborer du logiciel et du matériel, il est nécessaire de définir des interfaces pour interconnecter les composants : généralement le concepteur établit une norme pour tout composant et fournit un bus.

TABLE 3.6 – Les critères intervenants dans le choix du partitionnement

| Critère | Description |
|---|---|
| Coût | Critère omniprésent pendant tout le processus de conception du système. Sur de très petites séries, on va chercher à diminuer autant que possible le coût du développement du système logiciel et du matériel, tandis que dans le cas de grandes séries, c'est la diminution des coûts de fabrication qui revêtira une grande importance. |
| Performance | Son importance varie selon les problèmes traités. Les applications temps réel et pour lesquelles la robustesse est fonction du temps d'occupation processeur sont celles pour lesquelles ce critère est capital. C'est alors un partitionnement matériel qui sera à privilégier autant que possible. |
| Flexibilité | Ce critère joue en faveur du logiciel. Les effets de bords liés aux modifications du logiciel ont généralement un impact moins important sur le système global que dans le cas du matériel. La flexibilité des circuits logiques programmables effaçables s'améliore grandement. Ils sont aujourd'hui re-programmables in-situ, c'est-à-dire que l'on peut modifier leur programme sans les extraire de la carte électronique sur laquelle ils sont fixés. |
| Tolérance aux pannes internes | De par leur nature, les systèmes logiciels sont moins tolérants aux pannes. En effet, les micro-contrôleurs, sur lesquels ils reposent, utilisent des mémoires, des structures de piles sujettes au débordement, etc. Ce critère jouera en faveur d'un partitionnement matériel. |
| Dimensionnement | Il regroupe toutes les caractéristiques physiques du système (le poids, le volume, la consommation énergétique, le dégagement thermique etc.). Selon l'application, ce critère peut être hautement critique (cas des applications relevant des métiers de l'aéronautique). C'est au concepteur d'apprécier correctement ce critère. |
| Complexité algorithmique | Plus une tâche est complexe, plus il y aura de chance pour que l'on s'oriente vers un partitionnement logiciel. La synthèse matérielle des tâches hautement cognitives est par exemple beaucoup trop compliquée. |

La co-synthèse effectuée, il est possible de procéder à *la co-simulation* qui est la simulation simultanée des parties logicielles et des parties matérielles sans oublier leurs interactions afin d'analyser les propriétés dynamiques du système.

La dernière activité sera la *réalisation* du système. Chaque composant générique peut être implanté dans le matériel ou dans le logiciel comme l'illustre la figure 3.14.
Un composant matériel est décrit avec un langage de haut niveau prévu à cet effet (le Very high speed integrated circuit Hardware Description (VHDL) {Hsu et al., 1995} utilisé dans la communauté de la micro-électronique). On y voit sa description externe (Component description) et le début de sa description interne (Component architecture). Par *synthèse logique*, nous entendons le processus permettant de transformer la précédente description en un réseau de portes logiques. La *synthèse layout* met en œuvre le placement et le routage des éléments du circuit afin de réaliser le composant matériel.

Le fonctionnement des composants logiciels est exprimé via un langage de haut niveau (ici le langage C, facilement adaptable à un contexte "embarqué"). La compilation permet la traduction du programme en "langage machine" dont les instructions primitives dépendent généralement du processeur cible. L'étape d'assemblage permet la création du code binaire qui sera placé en mémoire et exécuté par un processeur.

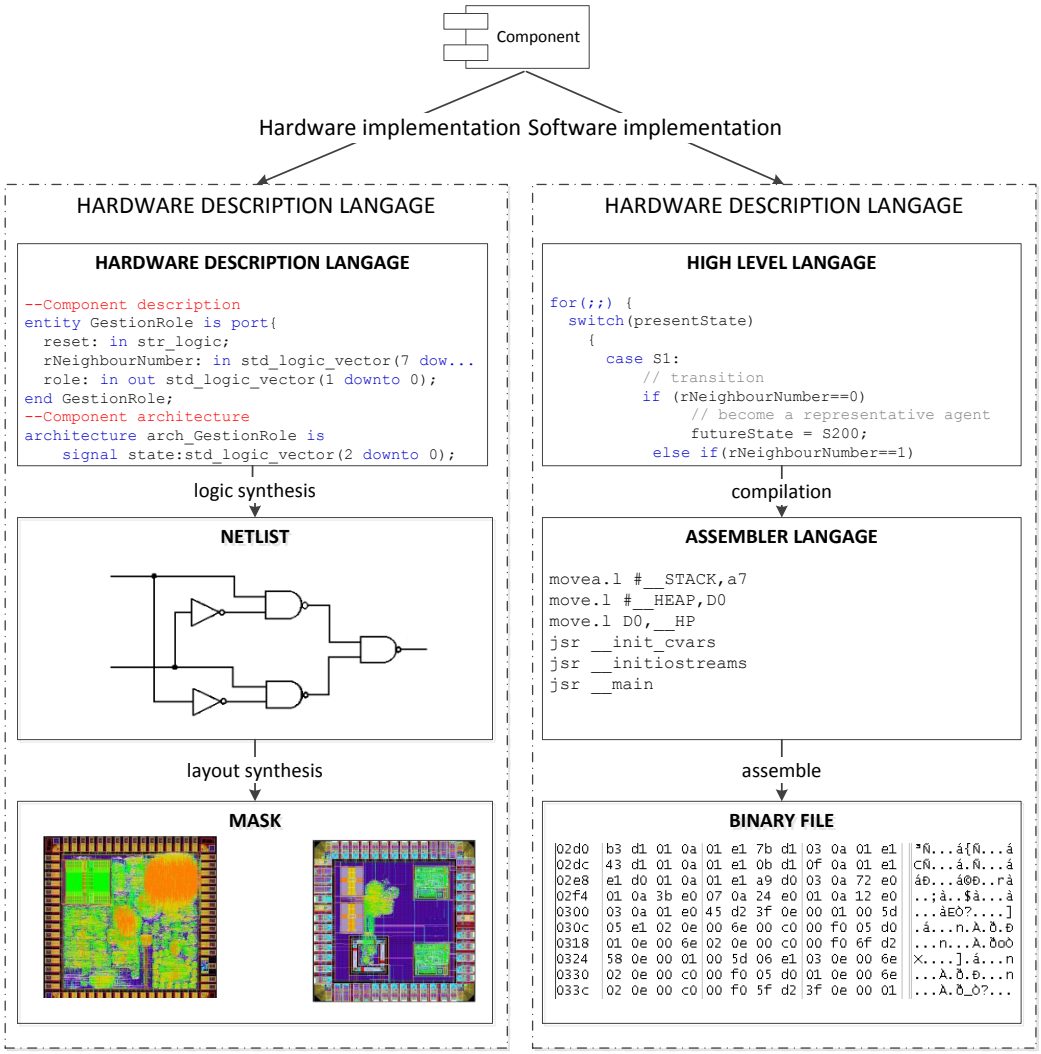



FIGURE 3.14 – Synthèse logicielle/matérielle des composants

Une illustration de la génération des spécifications matérielles des composants est donnée dans Jamont and Occello {2016}.

Cet aspect de la méthode a notamment bénéficié des discussions que nous avons eues avec Mouloud Koudil, spécialiste du co-design qui s'est notamment beaucoup intéressé au partitionnement Koudil {2002}.

## 3.6 Conclusion

Considérant les CCP comme des systèmes complexes artificiels, nous nous sommes intéressés aux apports possibles des SMA embarqués dans leur développement.

### 3.6.1 Sur l'adéquation des SMA pour concevoir des CCP

Nous avons identifié douze caractéristiques qui sont autant de défis que doivent relever les méthodes de conception. Notre motivation lors de l'élaboration de cet inventaire était de mesurer l'adéquation des approches, des modèles et des plateformes de la communauté multi-agent. Ces défis pourraient cependant être utilisés comme des critères d'évaluation de solutions multi-agents embarquées.

**Des défis naturellement pris en compte** Certains des défis (voir tableau de synthèse 3.7) sont naturellement pris en compte par les SMA que ce soit au niveau des méthodes, du paradigme ou des plateformes.

| | Méthode multi-agent | Paradigme multi-agent | Plateforme multi-agent |
|---|---|---|---|
| Réactivité (1) | | ⋆ | ⋆ |
| Terminaison (2) | | ⋆ | ⋆ |
| Ponctualité (3) | | ⋆ | ⋆ |
| Autonomie d'énergie (4) | | ⋆ | ⋆ |
| Gestion de la sécurité (5) | ⋆ | | |
| Complexité (6) | ⋆ | | |
| Concurrence (7) | ⋆ | | ⋆ |
| Hétérogénéité multiple (8) | | ⋆ | |
| Intégration (9) | | ⋆ | |
| Reconfiguation/Auto-organisation (10) | ⋆ | ⋆ | |
| Gestion de la mobilité (11) | ⋆ | ⋆ | ⋆ |
| Intégrité (12) | | ⋆ | ⋆ |

TABLE 3.7 – Défis relevés naturellement par les SMA

Au niveau du paradigme multi-agent, les capacités individuelles des agents permettent de répondre aux enjeux de la réactivité (1) ou de la ponctualité (3) qui sont liées à leurs aspects décisionnels. La gestion de l'hétérogénéité multiple (8) est liée à leurs capacités de gestion des connaissances et des comportements. La gestion de l'énergie (4) fait référence à l'autonomie décisionnelle des agents bien qu'elle ne soit cependant que rarement considérée comme un paramètre de leur décision au niveau des modèles. Garantir qu'un SMA va toujours atteindre un état satisfaisant est un problème d'autant plus difficile dans un environnement réel. Des travaux relatifs aux propriétés de terminaison (2) sont en cours et ils sont cruciaux pour la conception des CCP.
Les capacités sociales des agents permettent quant à elles l'intégration (9) qui est liée à la notion de coopération. La reconfiguration/l'auto-organisation (10), la gestion de la mobilité (11) et l'intégrité (12) sont liées aux notions d'interaction et d'organisation. L'intégrité (12) peut en plus bénéficier des modèles de gestion de la confiance et de la réputation. Les travaux sur l'intégration de systèmes multi-échelles (propriétés holoniques) peuvent aider le concepteur à mieux attaquer la complexité

(6) du système à concevoir. Enfin, la gestion de l'hétérogénéité (8) peut être traitée au niveau social comme un problème d'interopérabilité ou de couplage.

Au niveau méthodologique, la nature décentralisée de l'analyse participe à adresser la complexité (6) du système. La concurrence (7) est traitée via l'utilisation d'architectures évoluées d'agents.
La gestion de la sécurité (5), la concurrence (7), la reconfiguration et l'auto-organisation (10) et la mobilité (11) est prise en compte dans l'analyse des comportements des agents. Une attention particulière doit cependant être apportée à la gestion de la sécurité (5) en raison de l'importance du contexte physique. Des efforts doivent aussi être apportés sur les modèles concernant la ponctualité (3) en incluant la gestion du temps dans les modèles de raisonnement des agents.
La gestion de la mobilité (11) et de la confiance (Intégrité (12)) doit être impérativement introduite dans la phase d'analyse.

Concernant les plateformes multi-agents, elles doivent offrir la possibilité aux nœuds d'interagir avec le monde physique et d'être physiquement distribués.

**Des défis à mieux prendre en considération** Malgré tout, la communauté des SMA doit améliorer certains aspects pour couvrir l'ensemble de ces défis (voir tableau 3.8).

| Défi | Méthode multi-agent | Paradigme multi-agent | Plateforme multi-agent |
|---|---|---|---|
| Réactivité (1) | | | ⋆ Liée au monde physique |
| Terminaison (2) | | ⋆ Contraintes physiques | ⋆ Contraintes physiques |
| Ponctualité (3) | ⋆ Modélisation des besoins temporels | | ⋆ OS temps réel |
| Énergie (4) | | ⋆ L'énergie comme paramètre de la décision | ⋆ L'énergie comme paramètre physique |
| Sécurité (5) | ⋆ Modélisation des exigences de sécurité | | |
| Complexité (6) | | ⋆ Décision multi-échelle | |
| Concurrence (7) | | | ⋆ Architectures réellement décentralisées |
| Hétér. multiple (8) | | | ⋆ Interopérabilité |
| Intégration (9) | | | |
| Auto-org.. (10) | | | |
| Mobilité (11) | ⋆ Stratégies de gestion de la mobilité | | ⋆ Acceptation des réseaux mobile |
| Intégrité (12) | ⋆ Stratégies de gestion de la confiance | ⋆ Mécanismes de confiance et fiabilité | |

TABLE 3.8 – Défis qui méritent une amélioration des aspects multi-agents

Les défis les plus difficiles à traiter nous paraissent être ceux qui nécessitent une représentation explicite du temps (Défis 1, 2 et 3). Cet aspect est assez peu couvert, probablement parce que les applications utilisant des SMA embarqués sont encore trop peu nombreuses.

### 3.6.2 Sur nos contributions méthodologiques

Peu de travaux se sont penchés sur le problème de l'analyse des systèmes embarqués à intelligence décentralisée. Nous avons apporté quelques contributions pour répondre à certains de ces défis. Elles sont exploitées dans la méthode DIAMOND dédiée au développement de CCP.

Les spécifications évoluant généralement avec la compréhension du problème qu'ont les utilisateurs mais aussi les analystes, il est nécessaire d'accepter que les besoins soient remis en cause

pendant le projet. Le raffinement permet une exploration efficace de l'espace de conception afin de trouver par exemple un bon compromis logiciel/matériel. Le caractère incrémental d'un cycle milite pour la généricité. Nous avons adopté, pour ces raisons, un cycle de vie en spirale regroupant cinq phases qui exploitent des modèles et notations provenant de diverses sources (agents, composants, HDL etc.).

Dans la phase de recueil des besoins, nous avons introduit une étude des modes de marche et d'arrêt qui n'existe qu'à cause de la criticité de nos applications physiques. Très peu de méthodes multi-agents abordent les problèmes liés aux transitions d'un fonctionnement normal vers un arrêt, aux déclenchements d'arrêts d'urgence, aux arrêts pour maintenance, aux marches dégradées (que les entreprises exploitent pour maintenir les fonctionnalités malgré des dysfonctionnements) et surtout à la sécurité des biens et personnes.

Les phases de déploiement sont aussi assez marginales dans les méthodes multi-agents. Dans notre approche, cette phase transparaît à deux niveaux :

— au niveau de l'agent via la phase du partitionnement dans le sens où chacune de ses fonctionnalités peut être déployée sur du matériel et/ou du logiciel ;

— au niveau système dans laquelle chaque agent est déployé dans l'environnement physique conformément aux exigences exprimées durant le recueil des besoins.

Notre phase de conception est originale de par son côté générique. Elle fait l'abstraction de la nature logicielle/matérielle des agents.

La phase d'implantation démarre par la construction de la coquille externe des agents et se poursuit en direction de leur cœur (l'aspect décisionnel). Cette phase est effectuée sans discriminer ce qui deviendra logiciel ou matériel. On utilise pour cela des composants comme unité opératoire. L'analogie composant logiciel/composant électronique permet aux outils du logiciel et de la description du matériel de partager une vision unifiée.

La méthode DIAMOND est issue de réflexions théoriques sur la conception des SMA embarqués et d'une approche expérimentale des SMA : elle a profité de plusieurs retours d'expérience comme nous l'avons évoqué tout au long de ce chapitre.

# Chapitre 4

# Des modèles centrés individus pour les collectifs cyber-physiques

Le concepteur d'un CCP doit s'intéresser aux différentes facettes du système que l'on retrouve dans la décomposition AEIO. Les modèles pour le collectif qu'il utilisera ou qu'il développera afin de répondre aux attentes du système (ce qu'un agent ne peut faire seul) se focalisent sur les aspects *organisation* et *interaction*. Implanter le CCP consistera à intégrer dans les agents les différentes dimensions de l'analyse AEIO. L'utilisation d'une architecture d'agent adaptée aux contraintes de l'environnement facilite cette intégration.

## 4.1 Modèles d'agents

Nous définissons dans cette section les deux architectures d'agents que nous utilisons régulièrement pour mettre en œuvre nos contributions.

### 4.1.1 Le modèle eASTRO

Le modèle *ASTRO* a été proposé par Occello et al. {1998} pour intégrer des contraintes temps réel (défis 1, 2 et 3). L'architecture embedded ASTRO (eASTRO) est née de sa dégradation pour l'adapter à certaines spécificités liées au contexte embarqué, notamment l'autonomie énergétique des agents (défi 4). Il a été développé durant ma thèse et il est utilisé dans la plupart de nos déploiements de SMA embarqués.

Le modèle des agents eASTRO est basé sur le paradigme perception/évaluation/raisonnement/action. Le centre de l'agent est son modèle du monde qui contient ses connaissances sur l'environnement, sur les autres agents et sur son propre état. Cet état contient notamment les plans qui sont en cours d'exécution ou sur lesquels travaille le processus de raisonnement de l'agent. Ce modèle est rafraîchi par un processus d'interprétation des données perçues. Comme il évolue dans un monde dynamique, l'agent est muni de dispositifs de perception qui sont gérés par les modules de percep-

tion. Pour assurer la réactivité de l'agent, des évaluateurs examinent continuellement le modèle du monde. Les modules de contrôle de l'agent détectent les situations pour lesquelles l'agent doit réagir, les évaluent, et décident des réactions qui peuvent consister en l'introduction, la suspension ou la suppression de buts de façon à changer le contexte du processus de planification et de décision. La supervision continue de la situation de l'agent assure la réaction de l'agent à l'occurrence d'événements imprévus. Les modules de communication prennent en compte l'interaction avec les autres agents en utilisant si besoin des protocoles d'interaction.

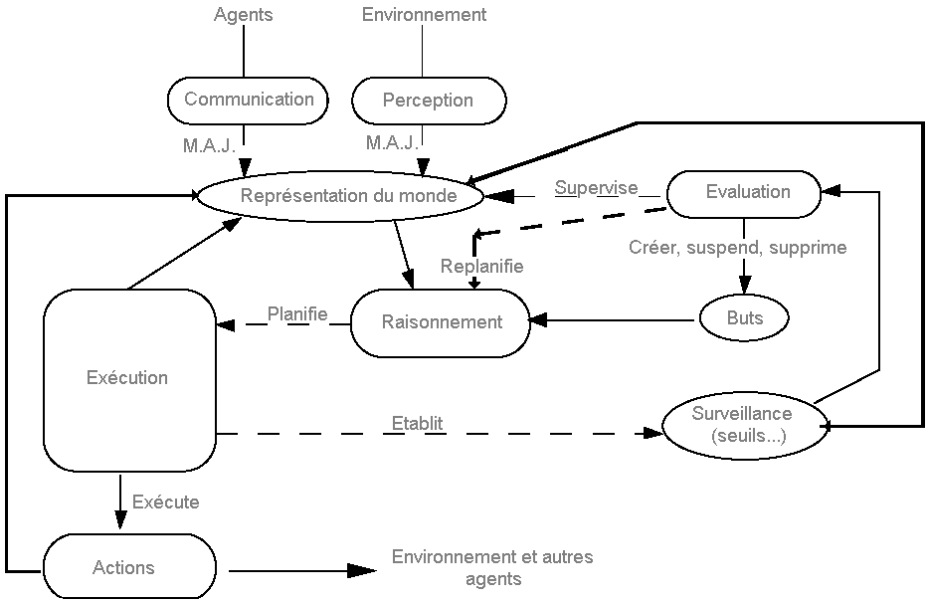



FIGURE 4.1 – Architecture eASTRO

Quand un but est créé (ou modifié), un plan est recherché par les modules de raisonnement pour le satisfaire en respectant une contrainte temporelle (date limite). Les plans construits sont ensuite ordonnancés par les modules de décision. L'ordonnanceur choisit la meilleure façon de réaliser le plan. Le planificateur et l'ordonnanceur ont la possibilité d'introduire des actions internes comme la replanification, ou l'établissement de conditions de veille pour garantir une meilleure adaptation au contexte. Ces gardes fournissent des informations sur la situation courante durant la phase d'exécution des actions. Ainsi, ce mécanisme assure l'adaptation de l'agent à la rapidité de l'évolution de son environnement ; il est indispensable si l'agent poursuit plusieurs buts simultanément.

Depuis ma thèse, cette architecture a été mise en œuvre dans le contexte de l'instrumentation sans fil d'environnements naturels (projet FITT EnvSys {Jamont et al., 2010}, projet PHC TASSILI {Hamani et al., 2011}), de la robotique collective (projet FITT Palette, projet Kurasu {Thomas, 2010; Jez, 2011}) et de la radiolocalisation Occello et al. {2008}. Pour ces projets, eASTRO a été déployé sur des plateformes généralement sans système d'exploitation (OS) à base de microcontroleurs Siemens C515C, Motorola 68HC12 et Microchip de la famille 16F.

### 4.1.2 Le modèle *Avatar*

Permettre l'interaction entre des agents purement logiciels et des agents physiques nécessite souvent d'utiliser des artefacts de programmation notamment pour :

— configurer les canaux de communication,
— assurer l'interopérabilité des messages,
— rendre les agents tolérants aux communications intermittentes,
— représenter visuellement les agents physiques pour simplifier la compréhension du collectif cyber-physique par un observateur artificiel ou humain.

Pour assurer la généricité des modèles, il est important d'externaliser ces artefacts. Pour que les agents physiques intègrent au mieux les contraintes extra fonctionnelles et que les organisations qui mettent en jeu des agents embarqués et des agents virtuels soient plus adaptatives aux problèmes de communication, il est nécessaire de donner une véritable autonomie de décision à ces composants logiciels. Ils deviennent alors des agents à part entière dont le rôle est de représenter les agents physiques dans le monde virtuel. Plus loin, ils peuvent étendre les capacités de perception et d'action des agents physiques. Le terme d'*avatar*[1] est alors particulièrement approprié pour désigner de tels agents.

Nous avons développé notre propre modèle d'*avatar* dans le contexte de nos activités sur le Web des Objets. Il est le résultat d'une collaboration initiée en 2010 avec Lionel Médini et Michael Mrissa (LIRIS) dans le contexte de nos participations au projet *Web Intelligence en Rhône-Alpes*. Elle a par la suite été élargie aux différents partenaires du projet ANR Adaptive Supervision of Avatar/Object Links for the Web of Objects (ASAWoO) que nous avons déposé et qui finance ces travaux (l'entreprise Génération Robots et le laboratoire IRISA). Le modèle vise à étendre le comportement et les connaissances d'objets inertes (un objet tagué RFID sans capacité de calcul) ou d'objets légers (disposant de faibles capacités de calcul). Les défis majeurs relevés par un tel modèle sont clairement l'hétérogénéité multiple (défi 8) et l'intégration d'interfaces (défi 9).

L'architecture d'un avatar (Figure 4.2) est composée de gestionnaires regroupés en modules {Jamont et al., 2014a; Mrissa et al., 2014}. Chaque gestionnaire prend en charge une préoccupation particulière de l'avatar et interagit avec d'autres composants à l'aide d'une Application Programming Interface (API) spécifique. Le module de base de l'architecture déploie les composants dans le framework via le *Component Deployment Manager* et fournit des composants de bas niveau que nécessitent d'autres gestionnaires pour raisonner ou effectuer des tâches. Chaque composant logique de l'architecture peut être exécuté soit sur l'objet physique soit dans le cloud.

Les agents avatars appartiennent à une infrastructure WoT qui est restreinte au niveau d'un réseau local pour des raisons de confidentialité. Cependant, l'infrastructure permet la communication inter-avatar ainsi que des interactions avec le Web externe à cet "intraweb".

Au moment de l'initialisation de l'avatar, le *Capability Manager* du module *Local Functionality Module* interroge le *Appliance Manager* pour récupérer les capacités de base que l'objet peut effectuer. Ensuite, le *Local Functionality Manager* découvre sémantiquement les fonctionnalités que l'objet peut obtenir en utilisant ces capacités. Pour ce faire, il interroge le raisonneur interne de

1. Emprunté à Philippe Gautier (Directeur de Business2Any) qui désignait ainsi l'intelligence logicielle associée à un objet physique dans un système d'information {Gautier and Gonzalez, 2011}.

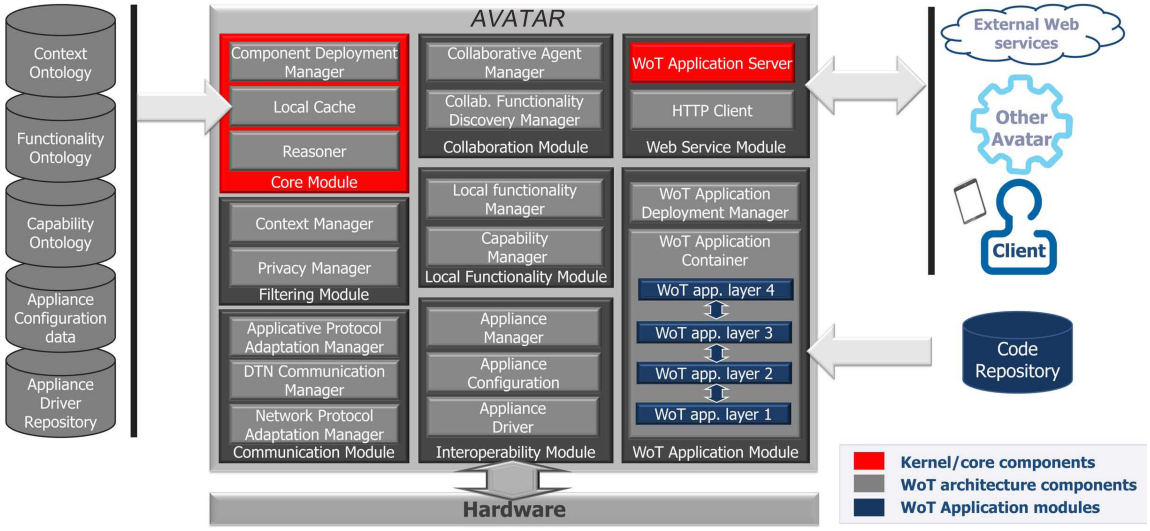



FIGURE 4.2 – Architecture d'un agent Avatar

l'avatar qui travaille sur l'ontologie de fonctionnalité. Certaines fonctionnalités sont complexes et impliquent plusieurs autres fonctionnalités pour être pleinement réalisées. Dans ce cas, un langage spécifique pour décrire l'orchestration de fonctionnalités est nécessaire. Par exemple, dans {Mrissa et al., 2014}, nous avons utilisé State Chart XML (SCXML). Les fonctionnalités découvertes sont filtrées à l'aide du *Context Manager* et du *Privacy Manager* afin de ne pas exposer celles qui sont irréalisables ou dangereuses que ce soit pour l'utilisateur ou les biens (défi 5).

Les fonctionnalités filtrées sont exposées comme des services aux utilisateurs et aux autres avatars par le serveur d'application WoT. Quand un service est invoqué, le moteur d'orchestration locale intégrée dans l'avatar déclenche l'invocation de services basés sur la description de l'orchestration de fonctionnalité. Pour chaque fonctionnalité "atomique" (ce qui est réalisé par une capacité), la mise en œuvre effective de la capacité est spécifique à chaque objet et stockée dans un dépôt de code de l'application.

Le code d'application qui implémente les fonctionnalités est exécuté dans un deuxième framework interne appelé *WoT Application Container*. Le code est dynamiquement déployé à partir du dépôt de code dans le framework au moment de l'exécution.

De la même façon, le *Collaborative Functionality Discovery Manager* découvre sémantiquement les fonctionnalités dans lesquelles l'objet peut être impliqué, mais qui nécessiteraient une collaboration avec d'autres objets. Il est interrogé par le *Collaborative Agent Manager* qui interagit avec d'autres avatars à travers le module *Service Web*. La composant *WoT Application Server* expose les fonctionnalités disponibles comme des ressources Representational State Transfer (REST). Chaque avatar utilise donc le module client Hypertext Transfer Protocol (HTTP) pour rechercher ou invoquer des ressources avatar dans l'infrastructure du WoT, ou interroger des services Web externes. L'avatar utilise également les modules d'interopérabilité et de communication pour se connecter respectivement à l'objet et pour communiquer avec elle en utilisant des protocoles réseaux les plus adaptés au contexte physique.

Deux étudiants de masters que j'ai co-encadrés avec Michael Mrissa ont permis d'implanter des

mécanismes décentralisés de coopération respectant les contraintes du Web (esquisse publiée dans {Khalfi et al., 2014}, version plus aboutie en cours de rédaction).

## 4.2 Des modèles pour les collectifs à communications contraintes

### 4.2.1 Motivations

**Contexte** Les nœuds d'un CCP communiquent entre eux en utilisant généralement des technologies sans fil dites *ad hoc* pour lesquels aucune infrastructure pré-existante et persistante n'est nécessaire. L'infrastructure [2] fait ici référence à l'ensemble des équipements qui permet d'assurer la connectivité des nœuds et de fournir les services réseaux de gestion des échanges. Les équipements de cette infrastructure sont par exemple les routeurs, qui gèrent l'acheminement des messages, et les points d'accès sans fil (Access Point (AP)) qui sont les points d'entrée des nœuds terminaux vers le réseau. Dans un réseau *ad hoc*, ces fonctions de gestion et d'administration du réseau ne sont pas centralisées sur des équipements dédiés mais assumées par l'ensemble des nœuds qui compose le réseau. Ainsi le déploiement et l'évolution d'un CCP ne nécessite ni la mise en place ni la configuration ni la maintenance d'une infrastructure réseaux.

La résolution décentralisée de problèmes et la réalisation collective de tâches nécessitent que les agents interagissent entre eux. Dans les SMA, ces interactions peuvent notamment reposer sur des appels de méthodes distantes ou via des communications directes c'est-à-dire nécessitant que les interlocuteurs soient mutuellement dans leurs portées d'émission. Cependant, la nature physiquement distribuée des CCP et les limitations des portées de transmission imposent souvent l'utilisation de communications non directes par sauts. De telles communications nécessitent la présence de nœuds relais pour acheminer la donnée d'une source au puit.

Le caractère multi-saut des échanges va entraîner l'utilisation de techniques d'inondation car, pour identifier les routes, aucun modèle complet du réseau n'est disponible : pour acheminer un message de la source au puits, la source envoie le message à tous ses voisins et ainsi de suite. Le coût d'obtention et de maintenance d'un tel modèle n'est en effet pas compatible avec les ressources limitées des nœuds. Cette recherche de route, qui nécessite de nombreuses communications, implique un coût important en énergie et une baisse possible de la qualité de service fournie utilisateurs aux exigences fonctionnelles. La qualité de service fait ici référence à la capacité du système de communication à respecter les exigences de l'utilisateur. Les critères de qualité de service traditionnels sont le temps de transmission de bout en bout, le taux de perte des paquets, la gigue, le taux d'erreurs binaires etc.,

La réalité physique de l'environnement et les technologies de transmissions utilisées rendent les communications asymétriques [3], limitent les bandes passantes et donc les débits.

Permettre aux nœuds du CCP de bénéficier des modèles d'interaction multi-agents nécessite d'abstraire ces problèmes de communication. Il n'existe cependant pas de solution de communication idéale et efficace dans l'absolu : elles sont toutes dépendantes de l'environnement électromagnétique,

2. Les infrastructures de réseaux sans fil sont souvent filaires (pour relier les différents points d'accès (AP) entre eux, connexion à Internet etc.).

3. Qu'un nœud $a$ puisse envoyer un message à un nœud $b$ n'implique pas que $b$ puisse envoyer un message à $a$.

du type de trafic (communication continue, en rafale...), des volumes à transmettre, des contraintes temporelles à respecter, des ressources d'énergie disponibles...

Soulager le concepteur d'un CCP des problèmes qui ne sont pas directement liés à son application est important en raison de la nature complexe du système qu'il doit concevoir pour répondre aux exigences fonctionnelles. Pour les non-spécialistes des réseaux, chacune de ces solutions est une boîte noire qui offre essentiellement une primitive d'envoi et de réception de messages mais surtout qui impose des choix technologiques pour chacune des couches réseaux. Par exemple, ZigBee impose une couche physique et une couche *Medium Access Control (MAC)* de type IEEE 802.15.4, le protocole de routage est *Ad-hoc On-demand Distance Vector (AODV)* etc. Pour une application donnée, le concepteur doit lui-même estimer la qualité de service (QoS) du service de communication et son impact sur le respect des exigences fonctionnelles et non-fonctionnelles qu'il a recueillies.

**Objectif et défis relevés** Pour exploiter toute la richesse des modèles disponibles dans le domaine des systèmes multi-agents et augmenter l'intégrité du système global, nous défendons l'idée qu'il est nécessaire de casser le caractère monolithique du service de communication. Cela permettra notamment de mieux relever les défis de la réactivité (Défi 1), de la ponctualité (Défi 3), de la mobilité (Défi 11), de l'intégrité (Défi 12) et de la gestion de l'énergie (Défi 4), en donnant plus d'autonomie décisionnelle à l'agent. Laisser en effet à l'agent la possibilité d'intervenir à tous les niveaux [4] du service de communication permet par exemple d'arrêter la transmission d'un message pour consacrer son temps de calcul à des tâches plus prioritaires (Défis 1 et 3), de connaître la liste des relais afin d'éviter que le message ne passe par des nœuds qu'il juge malveillants (Défi 12), etc. Globalement, la confiance (Défi 12) et la gestion de l'énergie (Défi 4) sont des contraintes fortes qui doivent être prises en compte à tous les niveaux de la conception et de la réalisation du CCP... et donc de la pile de communication.

Pour ce faire, nous ne proposons pas un nouveau protocole de gestion de message mais un modèle qui complétera la représentation du monde de l'agent et fera émerger une structure qui permettra de gérer de façon adaptative les communications avec une consommation énergétique raisonnée. Ce modèle sera opérationnel quelque soient les technologies des niveaux physiques et liaisons (couches généralement regroupées sous la dénomination MAC-$\varphi$).

### 4.2.2 Aperçu de la famille de modèles *MWAC

La famille de modèles que nous appelons *MWAC est constituée de MWAC et de 3 extensions :

— **Le modèle MWAC** {Jamont, 2005; Jamont and Occello, 2007; Jamont et al., 2010} modélise les nœuds du CCP par des agents. Il spécialise leurs rôles afin de faire émerger une structure adaptative aux changements de topologie. Il permet ensuite de trouver dans cette structure les chemins pour mettre en relation les différents émetteurs et les différents destinataires.
Les principaux défis applicatifs relevés par MWAC sont :
- réactivité (1) et ponctualité (3) : MWAC limite le volume total transmis durant les périodes d'inondation. De plus, il supprime une grande partie des envois périodiques de messages de découverte/vérification de voisinage inhérents aux approches traditionnelles. Il libère

4. On peut faire le lien entre les niveaux du service de communication et les couches du modèle OSI

ainsi du temps de calcul sur les nœuds ce qui augmente leur réactivité et peut améliorer la ponctualité de certaines actions.
- autonomie d'énergie (4) : limiter le volume transmis permet une économie de l'énergie.

— **L'extension TrustedMWAC** {Vercouter and Jamont, 2011, 2012} renforce le modèle MWAC en le dotant de mécanismes légers [5] de confiance qui pénalisent les fonctionnements non coopératifs des agents du système. Il permet ainsi de mettre en quarantaine des nœuds qui ont un comportement déviant, les excluant socialement du système.

> **Définition: Comportement déviant**
> Comportement qui constitue une violation de la norme qu'elle soit intentionnelle (cas du comportement malveillant) ou non (conséquence d'un dysfonctionnement).

Cette extension permet à MWAC de relever un défi applicatif supplémentaire :
- L'intégrité (12) est le défi relevé par ce modèle même si par effet de bord, et selon le type de l'attaque contrée, on peut estimer qu'il en traite d'autres (gestion de la sécurité (5), réactivité (1), etc.).

— **L'extension AntMWAC** {Hamani et al., 2011} diversifie la proposition de route du modèle originel afin d'anticiper les congestions. Cette extension est particulièrement intéressante quand les nœuds manipulent des flux continus d'informations ou qu'ils communiquent massivement pendant des fenêtres temporelles restreintes. Cette situation se produit généralement dans les systèmes pour lesquels la gestion de l'énergie est critique car les nœuds partagent alors souvent de longues périodes de sommeil et de courtes périodes d'activité. Cette extension permet à MWAC de renforcer certains des défis applicatifs :
- réactivité (1) et ponctualité (3) : Définissant deux classes de messages, AntMWAC affecte les chemins mis en évidence par MWAC aux messages les plus prioritaires et utilise la propriété d'exploration des fourmis pour les messages moins contraints en temps d'acheminement.

— **L'extension ≡MWAC** {Hoang et al., 2012} permet à un humain ou à un système d'aide à la décision, d'obtenir une vue intelligible d'un CCP implantant MWAC. Pour cela, il met en œuvre des mécanismes récursifs de composition qui diminuent la complexité du système global observé. Cette extension permet à MWAC de relever un défi applicatif supplémentaire :
- la complexité (6) car l'objectif de ce modèle est de créer des niveaux d'abstraction qui cachent les détails afin de mettre en évidence les phénomènes de plus haut niveau.

La figure 4.3 donne un aperçu de la structure d'un système dont les agents implantent MWAC et deux extensions (TrustedMWAC et AntMWAC).

La partie MAC-$\varphi$ est composée des dispositifs de communication qui assurent les communications de proches en proches. Elle inclut la couche physique et la couche liaison du modèle de référence OSI. Elle permet donc la transmission des signaux sur le médium de communication et le transfert de données entre entités à portée directe de transmission. Elle assure plus globalement la gestion de l'accès au médium, c'est-à-dire du "partage" du temps de transmission au niveau signal, l'évitement

5. Par léger nous entendons des mécanismes ne nécessitant pas d'infrastructure extérieure et ne consommant que peu de capacités de calcul et à faible empreinte mémoire etc.

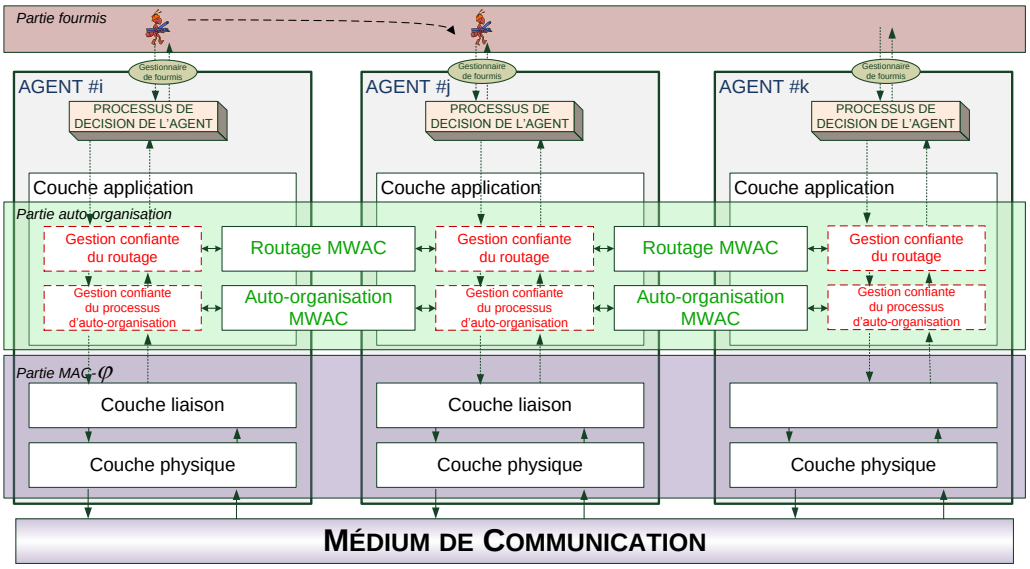



FIGURE 4.3 – Illustration d'une architecture de système mettant en œuvre MWAC et ses extensions TrustedMWAC et AntMWAC

de collision des trames émises, etc.

La partie auto-organisation permet la répartition du processus de routage du CCP sur tous les nœuds en respectant les structures émergentes inhérentes à l'utilisation de MWAC que les mécanismes de gestion de confiance ont rendu plus robustes. Un agent est ainsi capable de trouver une route vers le destinataire de son message.

La partie fourmis permet la mise en œuvre le modèle AntMWAC afin de trouver des routes faiblement utilisées en exploitant les capacités d'exploration des fourmis.

Les applications qui nous ont le plus intéressées jusqu'alors concernaient l'instrumentation des milieux naturels. Cependant, ces modèles peuvent être facilement adaptés à d'autres applications comme nous l'avons fait dans le cadre de la recherche de compétences dans les réseaux sociaux {Mercier et al., 2012}.

### 4.2.3 MWAC : Un modèle pour la gestion adaptive des communications dans les CCP

#### 4.2.3.1 Objectif

Les CCP sont des systèmes physiquement distribués composés de nombreux nœuds qui interagissent avec l'environnement et coopèrent. Afin de permettre cette coopération, il est nécessaire de doter le CCP de communications fiables respectant des contraintes non fonctionnelles entre les différents nœuds, même s'ils ne sont pas directement connectés...

Nous avons initialement développé ce modèle dans le contexte du projet *ENVironment SYStem (ENVSYS)* soutenu par par le Fond Incitatif de Transfert de Technologie (FITT) de la région Rhône-Alpes qui visait l'instrumentation de cavités souterraines. Le fait d'être sous terre rendait

la propagation des ondes difficile : les techniques numériques étaient encore peu utilisées bien que des systèmes de radiocommunication analogiques avaient vu le jour pour les secours spéléologiques. L'idée générale du projet visait à étudier la faisabilité d'un réseau maillé de capteurs à partir de cette couche physique existante. Les informations mesurées devaient être collectées à la sortie immédiate du réseau *via* une station de travail type PC qui en effectuerait le traitement.

Les réseaux de capteurs sans fil sont des cas particuliers de CCP rendus critiques à cause des fortes contraintes énergétiques. Notre objectif était de concevoir un modèle embarqué sur des systèmes fortement contraints. Par exemple, Florent Bouchy (élève ingénieur en 2005) a travaillé sur le démonstrateur qui exploitait une carte électronique incorporant un CPU cadencé à une fréquence de 4MHz et dont les capacités de mémoire étaient de l'ordre de 32Ko (données et code).

Lorsque nous avons commencé ces travaux {Jamont et al., 2002}, peu de contributions multi-agents traitaient de la problématique du gestion de l'acheminement de l'information dans les systèmes embarqués en réseaux.

#### 4.2.3.2 Description succincte du modèle

La construction des routes dans un réseau sans infrastructure pré-existante nécessite d'avoir recours à l'inondation. Le modèle MWAC permet d'acheminer des messages entre des sources et des destinations dans les réseaux sans fil tout en établissant une structure qui limite le nombre des connexions actives pendant le processus d'inondation. Pour cela, il spécialise le rôle des nœuds du système :

1. *Représentant* ($r$) : un nœud adoptant ce rôle prend en charge l'acheminement des messages que veulent transmettre les nœuds auxquels il est directement connecté. Pour ce faire, il diffuse, relaye et répond à des requêtes de recherche de route.
2. *Liaison* ($l$) : un nœud adoptant ce rôle assure la mise en relation des différents nœuds représentants auxquels il est connecté.
3. *Simple membre* ($s$) : un nœud adoptant ce rôle communique uniquement avec le nœud représentant auquel il est connecté. Il lui délivre les messages qu'il souhaite transmettre ou réceptionne les messages qui lui sont destinés.

**Agents.** Chaque nœud du CCP est modélisé par un agent. Soit $A_t = \{a, b, c, d, ...\}$ l'ensemble des agents du SMA à la date $t \in \mathbb{T} \equiv (\mathbb{N}, \leq)$ l'ensemble ordonné des dates.
Un SMA est modélisé sous la forme d'un graphe simple $g_t = (A_t, \omega)$ dont les sommets sont les agents et dont les arcs représentent leurs connexions.
Étant donné $a \in A_t$, on appelle le voisinage de $a$, $V(a) = \{b \in A_t \mid (a, b) \in \omega\}$.
Remarque : $a \in V(b) \Leftrightarrow b \in V(a)$

**Organisation.** Les agents spécialisent leurs rôles après analyse de leur voisinage. La structure ainsi obtenue privilégie l'utilisation de certains liens, ce qui permet de diminuer le nombre de connexions actives pendant l'inondation. On note $role(a)$ le rôle de l'agent $a$ $(role : A_t \rightarrow R)$ avec $R = \{r, l, s\}$ l'ensemble des rôles. L'organisation réalisée est caractérisée par les propriétés suivantes :

1. $role(a) = r$ ssi $\forall b \in V(a)$ $role(b) \neq r$

2. $role(a) = l$ ssi $\exists b \in V(a)\ \exists c \in V(a)\ \ role(b) = r$ et $role(c) = r$ et $b \neq c$
3. $role(a) = s$ ssi $\exists! b \in V(a)\ \ role(b) = r$

Cette organisation induit des observables que l'on appelle groupes. Un groupe est constitué d'un agent représentant et des agents de son voisinage. On remarque que chaque agent représentant détermine un groupe unique, que chaque agent liaison appartient à au moins deux groupes et que chaque agent simple membre appartient à un seul groupe.

La gestion des différents messages incombant aux agents représentants qui communiquent entre eux à l'aide des seuls agents liaisons, on définit $g'_t = (A'_t, \omega')$ le sous-graphe partiel de $g_t$ tel que :

— $A'_t = \{a \in A_t \mid role(a) \neq s\}$

— $\omega' = \{(a, b) \in \omega \mid a \in A'_t, b \in A'_t, role(a) = r \text{ ou } role(b) = r\}$

**Routes.** Lorsqu'un agent $a$ dont le représentant est l'agent $a_r$ veut communiquer avec un agent $b$ dont le représentant est l'agent $b_r$, l'agent $a_r$ doit identifier le chemin que devra emprunter le message pour atteindre l'agent $b_r$ (principe du *routage par la source*).

La recherche d'un chemin entre $a$ et $b$ dans $g_t$ revient à rechercher un chemin entre $a_r$ et $b_r$ dans $g'_t$ (chemin que nous appelons *route*).

**Algorithme.** Une implantation du modèle MWAC a notamment été proposée dans {Jamont et al., 2010}. L'algorithme peut être décomposé en trois étapes que nous allons illustrer à l'aide de la figure 4.4. Soit un CCP (figure 4.4a) modélisé par le SMA $g_t = (A_t, \omega)$, les étapes sont :

1. Établissement de l'organisation : Les agents spécialisent leur rôle afin qu'une organisation respectant les propriétés présentées ci-dessus soit établie (figure 4.4b).
2. Recherche de la route : Quand un agent représentant $a$ doit transmettre un message à un agent représentant $b$, il démarre une procédure de découverte de route qui consiste en la mise en œuvre du protocole DSR {Myles et al., 1995} sur les agents de $A'_t$ :
   (a) Un message `route_request` est créé par $a$ et inondé sur l'ensemble des arcs de $g'_t$ (figure 4.4c). La route est construite au fur et à mesure que le message se diffuse dans le réseau.
   (b) Chaque message `route_request` reçu par l'agent représentant $b$ correspond à une route découverte. L'agent $b$ choisit la route qu'il souhaite que le message utilise. Il le fait parvenir à l'agent $a$ via un message `route_reply`. Généralement [6], c'est la première route découverte qui est choisie car elle correspond, a priori, au chemin le plus court.
3. Transmission du message : L'agent $a$ génère et transmet un message `routed_data` qui contient la donnée à transporter et la route déterminée par $b$ qu'il faut utiliser (figure 4.4d).

#### 4.2.3.3 Discussion

**Adéquation au projet qui avait motivé ce travail** Nous avons dressé un comparatif (tableau 4.1) de notre solution avec des protocoles très populaires dans le monde des réseaux sans fil : Dynamic Source Routing (DSR), DSR-Routage et Destination-Sequenced Distance Vector (DSDV) en modulant les notes attribuées par des remarques critiques.

6. D'autres critères comme des valeurs de confiance associées aux agents peuvent amener à privilégier d'autres routes.

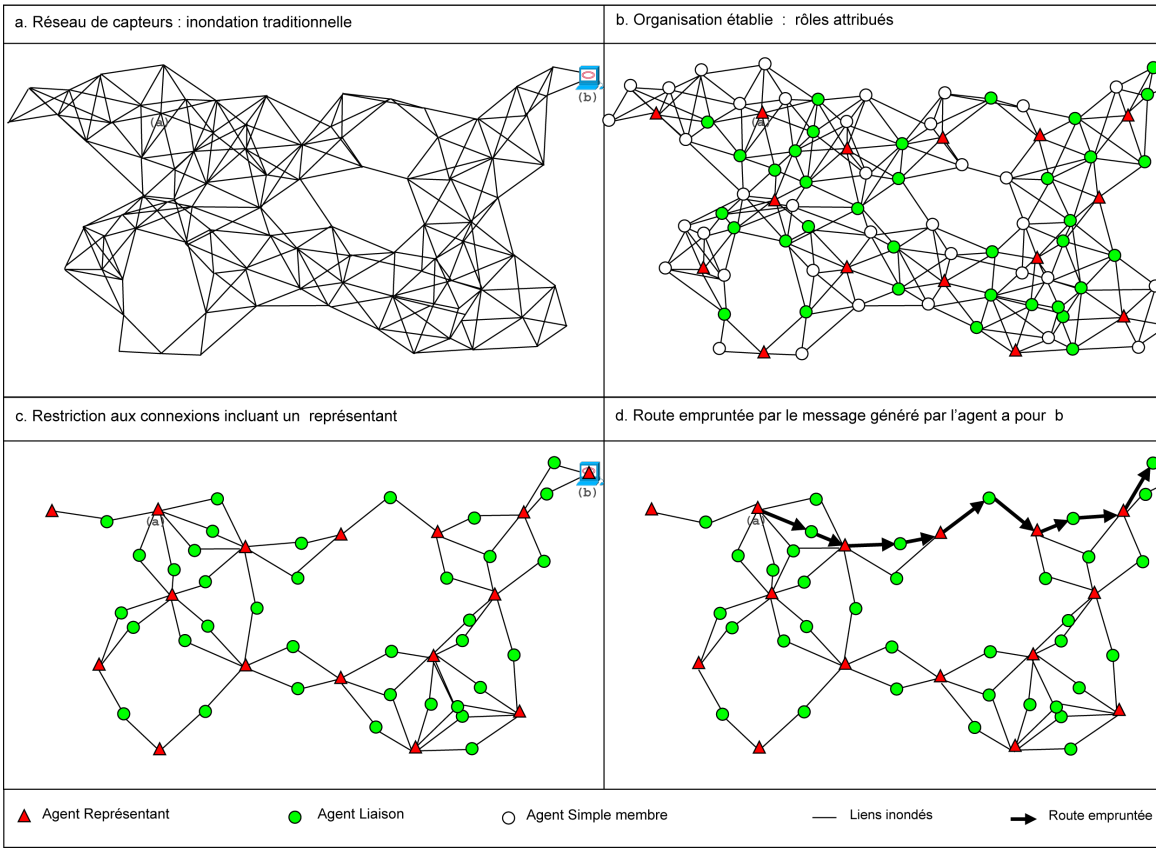



FIGURE 4.4 – Mise en œuvre du modèle MWAC

L'optimalité de notre solution est en termes de groupes : elle ne propose pas la meilleure succession de stations mais la plus courte succession de groupes.

La tolérance aux pannes concerne l'adaptation aux défaillances d'un lien suite à la mobilité ou à la destruction d'un nœud. Aucune solution ne peut être "meilleure" que DSR car elle est purement réactive : les routes sont recalculées à chaque émission de message.

Il est difficile d'apprécier le rendement protocolaire car il est fonction de divers paramètres tels que le volume utile des messages transportés (vu le caractère non optimal des routes proposées par notre approche, l'envoi de très grands messages pourrait être pénalisant), la fréquence d'émission des messages, du nombre moyen de stations dans l'aire d'émission/réception etc.

L'adéquation entre les attentes du projet EnvSys et les 3 autres solutions consiste en un classement.

**Évaluation du processus d'auto-organisation** L'appréciation d'un tel processus pour traiter un problème consiste à quantifier l'adéquation entre le processus d'auto-organisation conçu et les résultats espérés {Groupe MARCIA, 1996, 1997}. Il faut déterminer si le système est *valide* et *pertinent*. Ces deux propriétés seront corrélées avec *l'intérêt* et la *simplicité*. Le système est dit *valide* si le résultat produit par le processus auto-organisé est conforme avec ce que l'on attendait.

| Critére | DSDV | DSR | DSR-ROUTAGE | MWAC |
|---|---|---|---|---|
| Optimalité des routes | * * * * * | * * * * * | * * * * * | * * * |
| Tolérance aux pannes | * | * * * * * | * * * * * | * * * * |
| Rendement du protocole | Lié à la fréquence de rafraîchissement des tables. Mauvais rendement dans tous les cas. | Moyen | Moyen, bon dans le cas de protocoles unidirectionnels | Bon en général, très bon dans le cas de grands réseaux en grappes. |
| Adéquation pour EnvSys | 4 | 3 | 2 | 1 |

TABLE 4.1 – Tableau comparatif des performances des différentes solutions testées

L'*intérêt* est une grandeur qui estime la variation de cette *validité* en fonction de la complexité d'un raisonnement. Le système est dit *pertinent* si la structure peut être considérée comme simple pour ce qui est de la mise en œuvre et de la compréhension qu'en aurait un observateur. La *simplicité*, à l'image de l'*intérêt*, est une grandeur qui estime la variation de la *pertinence* en fonction de la complexité du raisonnement.

Cette étude a été réalisée dans {Jamont, 2005} et rapportée dans {Jamont et al., 2010}. Dans les applications visées, la structure de groupes des agents a donc bien permis une réduction du volume des messages qui permettent la localisation des autres agents et donc l'établissement de routes. La *validité* de notre approche avait donc pu être établie.

Les raisonnements mis en jeu par les agents qui participent au processus d'auto-organisation ne nécessitent pas de primitives très cognitives. Le résultat (les groupes) de ce processus étant facilement identifiable et compréhensible par un observateur extérieur, la *simplicité* du processus pouvait être constatée.

Il est difficile de prouver la convergence, la stabilité et la sensibilité d'une implantation de notre processus. Dans nos différentes expérimentations, nous la constatons au travers de simulation.

### 4.2.4 TrustedMWAC : Augmenter la robustesse aux actes malveillants

#### 4.2.4.1 Objectif

**MWAC : un modèle vulnérable** Si l'adoption d'une approche auto-organisée permet à MWAC de s'adapter dynamiquement à des pannes comme la disparition d'un nœud, une défaillance qui entraîne l'envoi de messages dont le contenu est faux perturbera la définition des rôles et donc des routes. L'agent MWAC, présumant que ses voisins se comportent comme attendu, il est donc vulnérable à l'exécution de codes incorrects sur un de ces nœuds que ce soit du fait de malveillances, de bogues ou de défaillances matérielles. C'est un problème d'autant que les CCP sont des systèmes ouverts et qu'il est donc facile d'introduire des agents malveillants dans l'optique de nuire au fonctionnement global du système.

La construction d'une organisation dans MWAC repose sur l'envoi de messages à ses voisins contenant, entre autres, l'identifiant de l'émetteur. Comme l'illustre le diagramme de la figure 4.5, quand il est créé, un agent envoie un message d'introduction afin de se présenter. Il découvre ses voisins de la même manière qu'il se déclare par ce message de présentation. Un message de présentation n'oblige

pas un agent qui l'entend à répondre, contrairement à la réception d'un message d'introduction. Le bon fonctionnement du processus d'auto-organisation repose donc sur l'échange entre les nœuds des triplets $< id, role, groupes >$ où $id$ est l'identifiant de l'agent émetteur du triplet, $role$ le rôle de l'agent émetteur et $groupes$ la liste de groupes (ou le groupe) auxquels appartient le nœud.

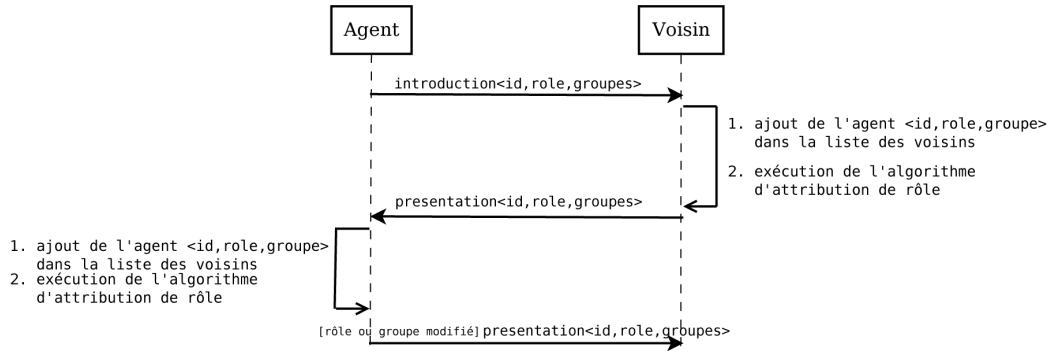



FIGURE 4.5 – Introduction dans la société d'agents

Une usurpation d'identité peut intervenir lorsqu'un agent envoie un message d'introduction (l'agent se présente à ses voisins pour s'introduire dans le système) ou de présentation (message permettant de faire connaître à un autre agent son triplet $< id, role, groupes >$ suite par exemple à la modification de l'environnement) avec un autre identifiant que le sien. Cette attaque crée une première perturbation sur l'acheminement des messages en déroutant vers l'attaquant ceux dont le chemin contient l'identité usurpée.

De plus, si un nœud multiplie les identités avec lesquelles il se déclare, cela peut amener à une saturation des tables de voisins et ainsi provoquer une faute de type déni de service (DoS). En effet, à chaque fois qu'un nouvel identifiant est perçu par un agent, il suppose qu'un nouveau voisin est présent et enregistre le triplet $< id, role, groupes >$ dans sa table des voisins.

La falsification du rôle ou des groupes associés à un agent n'est possible que si on usurpe son identité car les messages d'introduction et de présentation sont envoyés par l'émetteur et à destination de tous ceux qui peuvent comprendre le message véhiculé par l'onde électromagnétique. Au pire des cas, un agent malveillant peut créer une collision entre les messages pour qu'ils ne soient pas compréhensibles mais s'il veut faire parvenir de fausses informations sur un agent, il doit usurper son identité et générer un nouveau message. Les fausses informations qu'il peut diffuser portent sur :

- le rôle de l'agent usurpé, ce qui pourra entraîner des réorganisations locales car l'application de l'algorithme d'affectation de rôles par les agents qui perçoivent le message falsifié modifieront éventuellement leurs propres rôles ;
- les groupes associés à l'agent usurpé, ce qui provoquera des problèmes de routage car celui-ci est exprimé par une séquence de groupes à traverser pour atteindre la station de collecte.

Avec Laurent Vercouter (alors Maître Assistant à l'École des Mines de Saint-Etienne), nous nous sommes intéressés à la détection de mensonges dans la déclaration d'une identité à l'aide d'un modèle multi-agent de gestion de la confiance suffisamment léger pour pouvoir être déployé par exemple sur les plateformes au sein des réseaux de capteurs. Trois contraintes étaient donc importantes : un

fonctionnement décentralisé, une faible consommation des ressources et l'absence d'authentification.

**Décentralisation** Le domaine des systèmes multi-agents a proposé de nombreux travaux contribuant à la décentralisation des systèmes de gestion de confiance. L'intérêt est notamment de faciliter le passage à l'échelle en supprimant l'obligation de recourir à un tiers de confiance centralisé qui représente un goulet d'étranglement. Un nombre important de modèles de gestion décentralisée de confiance a ainsi été proposé (voir {Sabater-Mir and Vercouter, 2012} pour un état de l'art). Les valeurs de confiance sont calculées localement à partir de leurs propres perceptions mais aussi à partir de l'expérience d'autres agents. Le partage d'expérience nécessite des mécanismes particuliers afin d'estimer la crédibilité de celui qui la partage et d'éviter de considérer de fausses informations.

Les quelques travaux utilisant la confiance dans un contexte CCP comme celui des réseaux de capteurs que nous avions alors identifiés ({Boukerche et al., 2007; Han et al., 2007; Chen et al., 2008; Peng and Wei, 2008; Dai et al., 2009; Pietro et al., 2009; Ma, 2009}) utilisaient bien entendu des mécanismes décentralisés mais ne respectaient pas l'ensemble des contraintes imposées (comme l'absence d'authentification).

**Ressources limitées** Bien que la littérature académique présente des capteurs équipés de processeurs puissants et embarquant d'importantes quantités de mémoire (Sun Spot...), dans le monde industriel les capteurs réellement déployés ne disposent pas de telles ressources. Les raisons sont économiques (coût unitaire des capteurs) et énergétiques.

Les algorithmes de confiance qui utilisent généralement un historique d'observations et qui communiquent pour échanger ces observations doivent intégrer ces limitations. Beaucoup de travaux abordant la gestion de confiance dans les réseaux de capteurs ne prennent pas suffisamment en compte ce problème de ressources ou les repoussent à de futures perspectives comme dans {Babu et al., 2011}.

**Absence d'authentification** Tous les modèles de gestion de la confiance en système multi-agent supposent que les agents sont authentifiés. Dans ces travaux, il est en effet indispensable qu'une identité soit attachée à chaque agent et qu'elle ne puisse être contestée de manière à associer identités et valeurs de confiance. Il est généralement admis que des services d'infrastructure, tels que les *Public Key Infrastructure (PKI)*, soient utilisés pour assurer l'authentification. La très faible capacité de stockage et les contraintes de communication des capteurs rendent difficile le déploiement de telles infrastructures : il n'est pas possible que chaque nœud stocke les clés publiques de tous les autres, ni qu'il soit capable à tout moment de contacter un serveur central les conservant. Lewis et al. {2008}; Pei et al. {2009}; Liu and Zhou {2010} tentent cependant d'alléger ces mécanismes d'authentification mais cela reste insuffisant pour un déploiement sur des capteurs à faible capacité de mémoire. Fournel et al. {2007}; Castelluccia and Francillon {2008} présentent les principaux problèmes d'utilisation d'approches de cryptographie dans le contexte des systèmes contraints tels que les réseaux de capteurs.

L'absence d'authentification rend alors impossible l'usage de ces modèles de confiance. En effet, quand un agent reçoit un message, celui-ci peut provenir de n'importe lequel de ses voisins présents dans sa portée de communication, revendiquant n'importe quelle identité. Une gestion décentralisée de la confiance pour des CCP tels que les réseaux de capteurs pose ainsi un problème original

nécessitant l'adoption d'une nouvelle approche.
Peu de travaux prennent en compte l'absence d'authentification. Un des rares exemples est donné dans {He et al., 2008} qui propose une solution de gestion de confiance dans le routage d'information. Mais l'intérêt de cette proposition est difficile à quantifier car elle n'a été évaluée que par une simulation sur un réseau de cinq capteurs (ce qui laisse supposer des problèmes de passage à l'échelle au niveau de l'implantation du modèle).

#### 4.2.4.2 Présentation succincte du modèle

Ce modèle a notamment bénéficié du travail de Fouzia Hammouche (Master de Recherche de l'Université Jean Monnet, 2010) et de Anca Balanel (Master de Recherche de l'Université Jean Monnet, 2011) que nous avons encadrés avec Laurent Vercouter. Ces travaux ont permis d'explorer différentes pistes afin de fixer puis consolider les bases des réflexions exposées ci-dessous.

**Principe général** L'approche suivie dans le modèle TrustedMWAC consiste à utiliser la notion de confiance pour estimer la fiabilité d'un *voisinage* dans son ensemble plutôt qu'une évaluation indépendante de chaque voisin. Un voisinage indigne de confiance doit être interprété comme incluant au moins un agent malveillant ou défaillant. Lorsqu'un agent croit que son voisinage n'est pas digne de confiance, il adopte un fonctionnement *dégradé* n'effectuant que les tâches minimales attendues de lui. Ce fonctionnement dégradé doit éviter l'implication du voisinage de l'agent à toute tâche collective. L'objectif recherché par cette approche est que tous les voisins d'un nœud déviant détectent progressivement l'existence de mauvais comportements et passent en mode dégradé. La partie du réseau composé du nœud déviant et de ses voisins se retrouvera alors mis en *quarantaine* sans possibilité d'influencer le fonctionnement global du système. De plus, si l'écoute flottante est possible, un nœud pourra utiliser tous les messages émis dans son voisinage, et pas seulement ceux qui lui sont adressés, pour détecter d'éventuels mauvais comportements.

**Estimation de la confiance** Chaque agent réalise localement des estimations de confiance en observant les messages émis dans son voisinage. Même si l'authentification est impossible, les agents doivent utiliser un identifiant (véritable ou usurpé) lors de l'envoi d'un message. Nous avons proposé d'estimer la confiance d'un agent dans l'*usage d'un identifiant* plutôt que dans un agent authentifié par une identité.

*Initialisation et mise à jour* Une nouvelle valeur de confiance est créée à chaque fois qu'un nouvel identifiant *id* est utilisé dans le voisinage d'un agent. Cette valeur est comprise dans l'intervalle $[0; 1]$ avec pour valeur initiale une confiance maximale ($Trust(id) = 1$).
Si un message est jugé incorrect par rapport à un comportement communicatif attendu, nous parlerons de *mensonge* : il en découlera une diminution de la confiance dans l'identifiant utilisé.

*Détection de mensonge* Selon les cas d'observation, un mensonge peut être reconnu de manière sûre par un agent ou seulement suspecté. Dans le premier cas, la réduction doit être drastique car l'agent est assuré que son voisinage contient un nœud déviant. Dans le second cas, un événement inhabituel peut révéler un mensonge mais il existe quand même des cas exceptionnels où un tel message peut apparaître dans un fonctionnement normal. Par exemple, un nœud peut envoyer une information fausse s'il a une perception erronée de son voisinage qui l'amènera à affirmer appartenir

à un groupe alors qu'il vient de sortir du champ de communication du représentant du groupe. Si un mensonge est seulement suspecté, la confiance sera baissée avec une importance proportionnelle à la probabilité qu'une déviance en soit la cause.
Nous avons ainsi été amené à identifier les mensonges possibles dans MWAC, à trouver des mécanismes pour les détecter et proposer les actions à adopter. Ainsi, si un nœud perçoit un message utilisant son propre identifiant et au cas où un représentant perçoit un message venant d'un de ses voisins n'indiquant pas qu'il appartient au groupe du représentant, le message est indéniablement un mensonge visant soit à usurper l'identité d'un nœud, soit à cacher l'existence d'un groupe. La confiance est alors abaissée au plus bas niveau. Si un nœud qui n'était pas voisin de cette station entend un triplet dont l'identifiant est celui de la station de collecte [7], le message sera suspicieux (mensonge suspecté mais sans certitude) : les nœuds étant mobiles, cette situation peut tout de même se produire. Un voisin qui déclare appartenir à un nouveau groupe et éventuellement devenir nœud de liaison peut tout autant résulter d'une mobilité que d'une tentative d'altérer le routage. La lever de doute se fait alors collectivement. Le détail de l'ensemble de ces mécanismes est donné dans le papier de référence du modèle {Vercouter and Jamont, 2012}.

*Recouvrement de la confiance* Il est nécessaire que la confiance dans un voisinage puisse être rétablie. La mobilité, même faible, du réseau fait qu'un nœud malveillant peut être amené à quitter un voisinage. Il peut aussi être tout simplement retiré du réseau ou ne plus fonctionner faute d'énergie. Il se peut également qu'une défiance vis-à-vis d'un voisinage ne soit pas la cause d'une malveillance mais d'une co-occurrence d'événements exceptionnels ayant entraîné plusieurs suspicions. Le rétablissement de la confiance s'opère ici par un phénomène d'oubli avec une lente augmentation de la confiance avec le temps. Deux facteurs paramètrent la vitesse du rétablissement de la confiance : (i) la fréquence à laquelle l'algorithme de rétablissement est invoqué ; (ii) le taux d'évaporation de la sanction. La valeur de ces paramètres dépend principalement de la mobilité supposée des nœuds. En présence d'une mobilité forte, nous recommandons une fréquence et un taux d'évaporation importants car les voisinages devraient souvent varier. Si la mobilité est faible, le réseau sera plutôt statique et il est préférable de ralentir le rétablissement avec des valeurs plus faibles pour ces paramètres.

*Prise de décision de confiance* La confiance dans les identifiants est utilisée pour estimer la confiance du voisinage dans son ensemble. Le voisinage d'un nœud est jugé indigne de confiance s'il existe au moins un identifiant dans lequel le nœud n'a pas confiance.

**Intégration dans MWAC** L'objectif principal du maintien d'un modèle de confiance dans le voisinage d'un nœud est de sécuriser l'auto-organisation en empêchant les nœuds déviants de l'influencer. Lorsqu'un nœud n'a pas confiance dans son voisinage, il adopte un fonctionnement dégradé et ne participe plus au routage (sauf pour envoyer à son représentant ses propres mesures). Le mode dégradé correspond dans *TrustedMWAC* à un nouveau rôle *BACKUP* qui correspond à un fonctionnement similaire au rôle de simple membre, excepté qu'il est impossible d'évoluer vers un rôle de représentant ou de liaison.

**Propagation des fonctionnements dégradés** L'échange de recommandations et la construc-

7. Usurper l'identité de la station de collecte est très intéressant pour un nœud malicieux car le nœud a un rôle central d'un point de vue communication (elle collecte les données de l'ensemble des informations pertinentes pour l'application)dans la prise de décisions (la partie décisionnelle de l'application utilise ces données).

tion collective de réputations sont des pratiques classiques en gestion de la confiance pour augmenter le nombre d'informations prises en entrée et accélérer l'apprentissage de valeurs de confiance précises. Cependant, l'absence d'authentification nous empêche l'usage de ces techniques car elles nécessitent aussi d'associer une valeur de confiance ou de réputation à une identité indéniable.

L'échange de valeur de confiance ne présente dans notre cas que peu d'intérêt. Par contre, le fait de savoir que certains voisins d'un nœud ont adopté un rôle $BACKUP$ (mode dégradé) est utile. Cela signifie qu'il y a probablement un agent malveillant à proximité et peut-être dans le voisinage direct du nœud qui reçoit cette information.

Afin de partager cette information, les agents de *TrustedMWAC* prennent en compte l'éventuel rôle $BACKUP$ de leurs voisins. Un nœud qui informe ses voisins de son rôle (message d'introduction) ajoute les identifiants en lesquels il n'a pas confiance s'il est $BACKUP$. L'objectif est de propager ce message à tous les voisins présumés du nœud ayant cet identifiant afin que ceux-ci passent également en mode dégradé. Si tout le voisinage du nœud déviant est en mode dégradé, son entourage dans le réseau sera mis en quarantaine et il ne pourra plus nuire au bon fonctionnement du système.

La figure 4.6 illustre ce fonctionnement. Un agent malicieux est détecté par l'ensemble de ses voisins. Ils adoptent un fonctionnement dégradé qui a pour effet de mettre en quarantaine la zone dans laquelle est l'agent malicieux. Les liens entre les agents en mode dégradé et tout autre agent devient de fait un lien dégradé dans le sens où les agents prendront garde à ne pas véhiculer d'informations délicates.

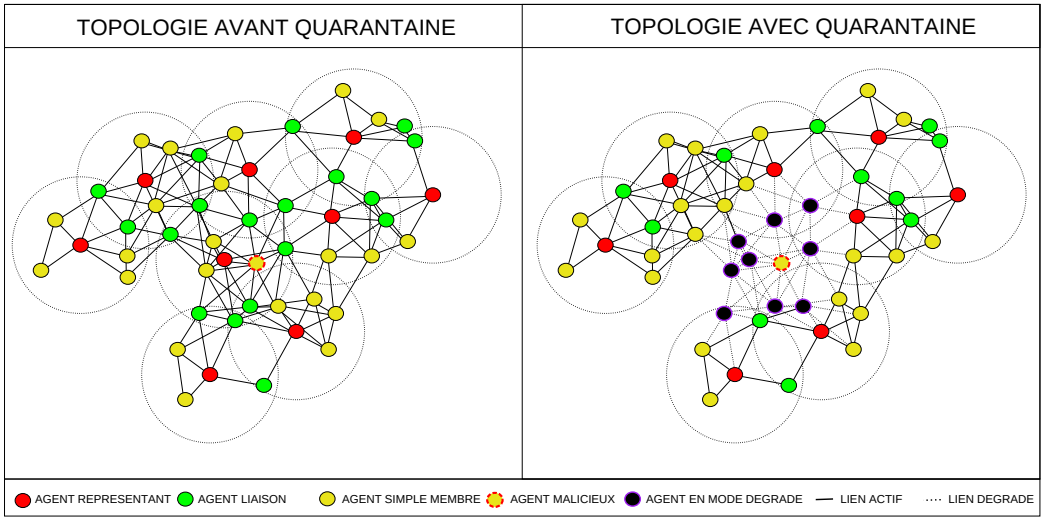



FIGURE 4.6 – Mise en quarantaine d'une région du réseau

Il est à noter que, pour éviter que la suspicion se propage à tout le réseau, les identifiants ayant provoqué le passage dans le rôle $BACKUP$ sont communiqués. Ainsi, seuls les nœuds ayant perçu un de ces identifiants dans leur voisinage passent en mode dégradé. La propagation s'arrête dès que l'on s'éloigne de ces identifiants.

#### 4.2.4.3 Discussion

La principale contribution de ce travail sur la confiance est un modèle ne nécessitant que très peu de ressources matérielles.

L'approche suivie repose sur un contrôle social pratiqué par l'ensemble des agents. Le principe est que chacun surveille qu'il n'y a pas d'anomalies dans les communications échangées dans son voisinage. Si c'est le cas, il se "sacrifie" en fonctionnant dans un mode dégradé, ce qui signifie qu'il continue d'effectuer ses fonctions de capteurs mais plus celles nécessaires au routage. Le réseau pourra alors se protéger d'un nœud malveillant si tous ses voisins, percevant des messages incorrects, passent en mode dégradé, créant une zone de quarantaine autour du nœud déviant. L'isolation d'une zone est rendue nécessaire du fait de l'absence d'authentification qui empêche l'identification précise d'un nœud.

Un mécanisme de propagation a été proposé pour étendre la détection d'anomalies à des nœuds qui n'auraient pas la capacité de les détecter (par exemple, parce que leur zone de perception des messages est insuffisante). Afin d'éviter que le mode dégradé ne se propage dans tout le réseau, la propagation doit indiquer quels identifiants sont indignes de confiance. Seuls les nœuds ayant un de ces identifiants dans leur voisinage prennent en compte la propagation en passant en mode dégradé et en faisant suivre le message à leur voisin.

Le risque d'un envoi d'une fausse propagation (par un nœud malveillant) existe néanmoins. Cette attaque aurait pour effet de mettre en quarantaine des zones valides. Elle peut cependant être contrée si l'on ajoute un autre cas de baisse de la confiance : il faut qu'un nœud injustement diffamé (c'est-à-dire qu'un message indiquant que son identifiant est indigne de confiance est propagé dans son voisinage) réplique en baissant la confiance de l'émetteur de cette diffamation à 0 et en propageant également cette information. L'effet de cette mesure serait d'étendre la zone de quarantaine qui inclurait alors les nœuds diffamés mais aussi l'attaquant. Si un recouvrement de la confiance est utilisé, les zones valides finiront pas sortir de la quarantaine après un certain temps.

Le surcoût de l'utilisation de la fonctionnalité "confiance" par rapport à MWAC réside essentiellement en deux points. Du point de vue de l'occupation mémoire, on ajoute une valeur de confiance à chaque voisin. Le triplet <`identifiant voisin,rôle,identifiant(s) groupe(s) associé(s)`> devient donc un quadruplet. Sur l'implantation réelle d'un agent MWAC en langage C, cela revient à ajouter une information de type `float` à la structure NeighboorInformation. Le code binaire d'une implantation en langage C sur micro-contrôleur Microchip 18F458 augmentera d'environ 20 %. Du point de vue de la charge en communications, le surcoût dépend de la malveillance (topologie réseaux, nombre d'agents malveillants, nature de l'attaque). L'utilisation d'une route permettant d'éviter une région du réseau augmente le coût de l'acheminement du message. L'appréciation de cette augmentation est à mettre en relation avec le bénéfice résultant du service rendu mais reste très difficile à estimer dans un cas général.

### 4.2.5 AntMWAC : Diversifier la proposition de routes

Le modèle AntMWAC est dédié aux applications des CCP dans lesquelles le caractère unidirectionnel des communications est prononcé comme dans le cas des réseaux d'instrumentation sans fil

(les messages partent des nœuds capteurs à destination des stations de collecte). Aussi, pour lever toute ambiguité nous parlerons de capteurs et non de nœuds comme dans les sections précédentes.

#### 4.2.5.1 Objectif

**Unicité des routes : une possible limitation de MWAC** Quand un nœud souhaite faire parvenir un message à la station de collecte, MWAC ne propose qu'une seule route. Pour certains domaines d'application des CCP comme la surveillance d'environnements naturels, cette unicité ne pose généralement pas de problème, le trafic est globalement faible. Cependant, pour certaines applications, le volume des données échangées peut ponctuellement devenir important. C'est par exemple le cas des systèmes véhiculant des flux vidéo {Du et al., 2008; Garcia-Sanchez et al., 2011}. L'utilisation de routes uniques par différents émetteurs peut amener à des congestions. Elle proviennent de la saturation des files d'attentes des nœuds qui composent la route. Les deux principales conséquences néfastes pour le système sont l'augmentation du temps d'acheminement des messages et l'augmentation du taux de perte des messages. La perte de messages est ici liée au rejet des messages par les agents dont les files de réception de message sont saturées.

**Approche traditionnelle** Pour pallier l'unicité de la route proposée par MWAC, plusieurs solutions sont possibles. Dans les réseaux, l'approche classique à ce problème consiste à découvrir et maintenir plusieurs routes {Wang et al., 2001; Lee and Gerla, 2001; Marina and Das, 2002} afin soit de répartir la charge de communication (évitement préventif de congestion), soit de les utiliser quand la route principale est non disponible (correction de la rupture de routes) ou qu'elle est encombrée (correction du problème de congestion lorsqu'il est détecté). Implanter ce type de solution dans MWAC nécessite de définir de nouveaux messages pour rechercher des routes qui évitent certains représentants. D'une part, utiliser cette solution de manière préventive, alors que l'unicité de la route ne provoque pas de problème, conduit à gaspiller des ressources et, d'autre part, utiliser cette solution lorsque le problème apparaît amplifie temporairement la charge du réseau.

**Une solution à base d'ACO** Les algorithmes de routage dans ce type de réseaux doivent proposer des routes qui optimisent différentes fonctions de coût liées à des critères de qualité de services tels que le temps d'acheminement des messages, la consommation énergétique...
Les méthodes d'optimisation que l'on peut utiliser quand la solution optimale ne peut être obtenue avec des conditions raisonnables (temps, ressources matérielles...) sont des méthodes d'approximation que l'on qualifie d'heuristique. L'optimisation par colonie de fourmis (Ant Colony Optimization (ACO)) en est un exemple inspiré du comportement réel des fourmis qui sont des insectes capables de résoudre des problèmes comme la recherche d'un plus court chemin. Les mécanismes de coordinations qu'elles utilisent sont en fait assez simples à modéliser. En effet, les fourmis se déplacent dans leur environnement à la recherche d'une source de nourriture dans un premier temps aléatoirement. Elles laissent sur leurs passages des marqueurs dans l'environnement (les phéromones) formant ainsi des pistes qui pourront être utilisées par d'autres fourmis.
Les fourmis logicielles ont des capacités d'interactions limitées et un système cognitif très simple. Si on modélise l'environnement par un graphe, une fourmi peut se déplacer d'un sommet vers un sommet voisin, accéder aux informations portées par le sommet où elle est située, et déposer une

quantité de phéromones sur l'arc qu'elle utilise pour se déplacer.
A l'origine du modèle AntMWAC se trouve la conviction que les ACO sont une bonne technique d'optimisation de routage. Le problème du routage peut être en effet décrit comme un problème de recherche de plus court chemin sur un graphe dont la métrique serait la fusion de plusieurs critères de QoS. Ensuite, l'utilisation de faibles ressources par les fourmis fait écho aux limitations matérielles des capteurs. Enfin, le processus de décision des fourmis n'utilisant que des informations disponibles dans son environnement local (il n'est pas nécessaire d'avoir une vue globale du graphe), il est adapté pour les systèmes fortement distribués tels que les réseaux de capteurs.
L'originalité de notre approche, par rapport aux autres utilisations des ACO dans le contexte du routage {Huang et al., 2010; Yang et al., 2010; Yanrong and Hang, 2010}, est que le problème ne consiste pas à identifier les chemins les plus courts du réseau mais à trouver des routes alternatives, éventuellement plus longues, sans inonder le réseau de messages de recherche de routes alternatives.

#### 4.2.5.2 Présentation succincte du modèle

Ce modèle est une des contributions de la thèse de Nacer Hamani (enseignant à l'École nationale Supérieure d'Informatique d'Alger et dont je co-dirige la thèse avec Pr. Mouloud Koudil).

**Fonctionnement de l'optimisation par colonie de fourmis** Une approche d'optimisation par colonie de fourmis est constituée de plusieurs étapes :

*Initialisation* Phase durant laquelle la distribution initiale des fourmis sur les nœuds est effectuée, et les liens entres les nœuds sont initialisés avec une quantité $\tau_0$ de phéromones.

*Construction des routes* Phase d'exploration pendant laquelle une fourmi se déplace d'un nœud à une autre choisi à l'aide d'une règle de transition d'état. Cette règle utilise une heuristique et l'attractivité des arcs dont le départ est le nœud qui l'abrite. Plus un arc supporte une quantité importante de phéromones, plus elle est attractive.
Soit $k$ une fourmi située sur un nœud $i$. Soit $N_i$ l'ensemble des nœuds voisins de $i$. Soit $M_k$ l'ensemble des nœuds déjà empruntés par $k$. Soit $\tau_{ij}$ la quantité de phéromones sur l'arc $(i, j)$. Soit $\eta_{ij}$ une heuristique associée à ce même lien. On peut alors définir $p_{ij}^k$ la probabilité qu'une fourmi $k$ située sur le nœud $i$ se déplace vers un nœud $j$.

$$p_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha . [\eta_{ij}]^\beta}{\sum_{l \in N_i} [\tau_{il}]^\alpha . [\eta_{il}]^\beta} & j \in V(g_t, i) \text{ et } j \notin M^k \\ 0 & \text{sinon} \end{cases} \tag{4.1}$$

où $\alpha$ et $\beta$ sont deux paramètres qui contrôlent l'influence de l'heuristique et des phéromones dans le choix du prochain lieu visité. Les quantités de phéromones sont généralement des valeurs réelles positives que l'on attribue aux liens entres les capteurs.

*Renforcement des routes* Les fourmis qui permettent la construction des routes sont appelées fourmis *forward*. Quand elles arrivent sur le lieu de destination, des fourmis *backward* vont retracer la route construite dans le sens inverse afin de renforcer la piste de phéromones : elles déposent une quantité supplémentaire de phéromones qui est fonction de la qualité de la solution trouvée.

*Évaporation* Les pistes de la phéromones vise à la réutilisation des expériences positives des autres fourmis. Cependant, des expériences négatives donnent aussi lieu à la création de pistes.

L'évaporation permet la dissipation de ces dernières.

*Diversification et intensification* Les paramètres $\alpha$ et $\beta$ (équation 4.1) permettent de réguler l'influence à la fois de la fonction heuristique et des pistes de phéromones dans le choix du prochain nœud à visiter.

Si $\alpha \gg \beta$, on favorise l'importance de la quantité de phéromones dans la règle de transition, les fourmis privilégieront l'utilisation de pistes de phéromones plutôt que la découverte de nouvelles : on parle alors d'un phénomène d'*intensification*. Une conséquence néfaste pour le système global est l'apparition de la stagnation, nom donné au fait que les fourmis construisent toutes les mêmes routes. Il s'en suit de possibles congestions.

Si $\beta \gg \alpha$, on favorise l'importance de l'heuristique dans le processus de décision des fourmis. Les fourmis vont donc explorer le graphe pour éviter autant que possible les pistes de phéromones existantes : on parle alors de *diversification*. L'effet néfaste est l'amoindrissement (voire l'annulation) de l'effet collaboratif.

**Notre approche** D'un point de vue architectural, l'optimisation par colonie de fourmis est généralement une couche générique qui vise l'amélioration de la qualité des résultats fournis par la couche de routage sous-jacente. Cette couche de routage permet de guider les fourmis pendant la phase exploratoire en leur proposant une direction initiale. Par exemple, Zhu {2007} utilise les fourmis pour améliorer la répartition de la consommation énergétique dans le protocole Direct Diffusion {Intanagonwiwat et al., 2000; Zhang et al., 2004; Goel and Sharma, 2009} améliorent respectivement les performances des protocoles AODV {Perkins et al., 2003} et ADR {Lu et al., 2004}

Plus qu'une implantation d'algorithme d'optimisation par colonie de fourmis sur MWAC, notre extension met en interaction les agents et les fourmis. Ainsi, les données transportées par les fourmis peuvent, par exemple, donner aux agents des informations sur leur voisinage. Les agents peuvent quant à eux guider les fourmis en modifiant leurs paramètres décisionnels afin qu'elles adaptent leurs comportements à l'environnement. Les tables de phéromone traditionnelles sont distribuées sur les agents qui les gèrent.

Nous présentons ci-dessous l'heuristique que nous utilisons et les différentes interactions entre fourmis et agents. .

**Heuristique** Pour anticiper l'apparition de congestions dans le réseau, les fourmis doivent, à partir de la proposition de routes de la couche auto-organisation, trouver des routes (dans $g_t$ — cf formalisation de MWAC page 55) qui évitent autant que possible d'utiliser les nœuds déjà empruntés (dans $g'_t$). Pour ce faire, le taux d'occupation des liens est intégré dans la fonction heuristique de manière à pénaliser les liens les plus utilisés. Afin de répartir la dépense énergétique plus équitablement qu'avec MWAC, nous intégrons aussi l'énergie résiduelle d'un agent dans cette heuristique.

**Production de fourmis** Les agents MWAC gèrent deux types de messages : les *messages de configuration* pour construire l'organisation et maintenir sa structure et les *messages de transport de données*. Afin de pouvoir adapter la transmission des messages applicatifs vers la station de collecte aux congestions, sans les amplifier, nous définissons deux classes de services pour le transport des données :

— La *classe 1* repose sur l'utilisation stricte des routes proposées par MWAC, indépendamment

du travail des fourmis. Le message est alors acheminé via une succession optimale de groupes. Les agents MWAC peuvent a priori connaître le temps moyen d'acheminement d'un message en fonction des expériences de transmission précédentes.

— La *classe 2* repose sur l'utilisation des fourmis. Aucune garantie n'est alors donnée sur le temps d'acheminement des messages. A chaque envoi, l'agent crée une fourmi de type *forward* qui véhicule la donnée. Le destinataire final, après réception du message, procédera à la création d'une fourmi *backward* qui permettra à l'émetteur initial d'obtenir un accusé de réception et d'estimer le temps d'acheminement.

Quand le réseau n'est pas congestionné, toutes les transmissions utilisent la classe 1. Quand un agent détecte des congestions (dérive positive du délai d'acheminement), il n'utilisera cette classe que pour les messages prioritaires [8]. Si le message n'est pas considéré comme prioritaire, il confiera l'acheminement des données aux fourmis (classe 2).

**Orientation initiale des fourmis** L'orientation des fourmis est d'une manière générale donnée par la quantité de phéromone sur des liens avec les nœuds voisins. Cependant, à l'initialisation du réseau aucune piste de phéromone n'existe. Les fourmis explorent donc l'espace de recherche pour trouver la station de collecte sans privilégier de direction.

Afin que ces fourmis trouvent des chemins différents de ceux proposés par MWAC, il faut que les quantités de phéromone soient plus importantes sur les liens qui sont les moins sollicités. Les agents simples membres ne sont aux extrémités d'aucun des liens d'un chemin proposés sur MWAC contrairement aux agents liaisons et représentants. La quantité $\tau_r^S$ qui est ajoutée à un lien à destination d'un simple membre sera donc la plus importante. Un agent liaison étant tout de même moins sollicité qu'un agent représentant, la quantité $\tau_r^L$ associée à lien vers un agent liaison sera plus grande que $\tau_r^R$.

**Équilibre dynamique entre l'intensification et la diversification** Les approches classiques de l'ACO fixent les valeurs de $\alpha$ et $\beta$ de la règle de transition (équation 4.1) au début de l'algorithme. Elles sont choisies essentiellement en fonction de la topologie du réseau et de l'environnement physique. Dans notre approche, nous faisons l'hypothèse des environnements dynamiques. L'agent, à partir de sa perception locale de son environnement, doit pouvoir modifier à tout moment les valeurs de ces deux paramètres afin de privilégier l'intensification ou la diversification. L'objectif de l'agent est de trouver un bon compromis entre l'utilisation des routes (ou des fragments de routes fiables), au risque d'épuiser rapidement l'énergie des nœuds qui la composent, et la diversification qui peut mener à utiliser des routes plus longues.

Ainsi l'implantation d'un diagramme de transition d'état {Hamani et al., 2011} permet aux agents d'adapter localement le comportement de la fourmilière aux conditions de l'environnement. Si par exemple un agent considère que son environnement est perturbé (nombre des trames reçues erronées et/ou le nombre de retransmissions de ses propres trames au-delà des valeurs habituelles), il modifiera les valeurs des paramètres afin que les fourmis réutilisent au mieux les expériences passées. Si un agent considère que son voisinage est instable (mobilité importante), il privilégiera la diversification car les pistes de phéromone ne traduisent plus obligatoirement l'existence d'un chemin.

8. Dans un réseau d'instrumentation, elle est généralement fixée par le concepteur en associant des valeurs à des types de messages ou à l'aide de règles.

**L'estimation de l'énergie** Lors de la construction d'une route, une fourmi doit calculer la probabilité de son déplacement sur chacun de ses nœuds voisins. Le calcul de cette probabilité fait intervenir le niveau d'énergie de ses voisins. Les nœuds devraient donc soit diffuser périodiquement leurs niveaux d'énergie à leurs voisins, soit répondre aux interrogations des nœuds voisins *à la demande*. Ces communications sont globalement coûteuses en énergie.

Dans le modèle AntMWAC, nous enrichissons les capacités des agents avec un module permettant d'estimer le niveau d'énergie des nœuds voisins afin de diminuer les émissions de messages véhiculant ces quantités d'énergie disponibles. Cette estimation ne prend en compte que la consommation d'énergie liée aux communications et ignore donc les autres coûts ce qui suppose une certaine uniformité du matériel.

Un agent $i$, qui estime le niveau d'énergie de son voisin $j$, doit pour cela connaître le volume à la fois des messages envoyés et des messages reçus par ce voisin. Via un processus d'écoute indiscrète, il connaît précisément [9] le volume des émissions de son voisin. Par contre le volume des messages reçus ne peut qu'être estimé car $i$ ne perçoit pas nécessairement tous les messages reçus par $j$ ($i$ n'est pas dans le voisinage de tous les voisins de $j$).

Étant donné que les agents sont homogènes d'un point de vue "communication" car ils utilisent AntMWAC, $i$ peut tout de même faire des suppositions réalistes sur les messages reçus par l'agent $j$ (et leur volume) en donnant un sens à la réponse faite par $j$. Par exemple, quand l'agent $i$ entend un message `route_reply` envoyé par son agent voisin $j$, c'est soit que $j$ a répondu à un message `route_request`, soit qu'il a relayé un message `route_reply`. Dans cet exemple précis, l'agent $i$ peut estimer le nombre de bits reçus par $j$ du message qu'il n'a pas perçu à 4 octets près.

Cette simple estimation, couplée à une diffusion opportuniste du niveau d'énergie, nous permet de supprimer l'envoi périodique des niveaux d'énergie ou de mettre en place un mécanisme d'interrogation/réponse.

**La distribution des tables de phéromone** Dans les applications centralisées utilisant les ACO, une table contient la quantité de phéromone associée à chaque arc du graphe sur lesquels les fourmis se déplacent. Dans notre contexte, il est nécessaire de distribuer physiquement cette table de phéromone : chaque nœud mémorise la valeur de phéromone des arcs à destination de ses voisins. Ainsi, modifier la valeur de phéromone associée à l'arc $(i, j)$ revient à modifier la valeur portée par l'arc dans la table de phéromone hébergée par le nœud $i$ et celle portée par le nœud $j$. Notre solution ne propose en cela aucune particularité par rapport aux autres implantations distribuées d'ACO.

**Vérification des connaissances de l'agent** Lorsqu'une fourmi part d'un agent $i$ pour rejoindre l'agent $j$, elle apporte à $j$ des informations. Cela permet à l'agent $j$ de vérifier et mettre à jour certaines de ses connaissances. L'utilisation des fourmis renforce donc l'écoute indiscrète et permet ainsi, par exemple, la diminution du coût de maintien de la validité de l'organisation.

Parmi les informations éventuellement transportées par la fourmi voyageant de l'agent $i$ à $j$, nous pouvons lister à titre d'exemple :

— l'identifiant du nœud $i$ : l'agent $j$ peut ainsi détecter l'apparition de l'agent $i$ dans son voisinage et permettre de déclencher l'adaptation de l'organisation à un changement de topologie,

9. Hypothèse des communications symétriques.

— le nombre de sauts déjà effectués par la fourmi : si la fourmi est de type "forward" et que l'agent sur lequel est la fourmi estime qu'il n'y a pas de congestion, il peut alors changer la classe du message. L'acheminement du message est donc confié à MWAC.
— l'énergie disponible du capteur $i$ : l'agent $j$ peut corriger la fonction d'estimation de l'énergie disponible de $i$. Cela consiste au remplacement de la quantité d'énergie estimée par la valeur réelle communiquée.

#### 4.2.5.3 Discussion

L'acheminement de messages dans un réseau d'instrumentation sans fil est parfaitement pris en charge par MWAC, qui propose à chacun des nœuds une route unique pour transporter leurs données vers la station de collecte. Néanmoins, selon le type d'application, des congestions peuvent apparaître : la saturation des files de réception peut alors augmenter significativement le temps d'acheminement des messages ou entraîner leur perte. Pour répondre à ce problème nous avons proposé une extension de MWAC qui permet de diversifier la proposition de route : le modèle AntMWAC. Pour cela nous définissons une classe additionnelle de services, exploitant la capacité de diversification qu'offrent les fourmis pour l'acheminement des données. Les agents MWAC interagissent avec les fourmis et deviennent des médiateurs entre l'environnement et les fourmis : ils modifient les paramètres de la règle de transition des fourmis pour aider la fourmilière à mieux adapter son comportement global aux conditions locales de l'environnement.
L'implantation de ce modèle passe l'échelle sur les différents scénarios que nous avons simulés (50 à 500 agents). Son utilisation a certes un coût, mais la dégradation de la qualité de service fourni aux utilisateurs est moins impactée.

Comme le bon fonctionnement global du modèle AntMWAC dépend du bon comportement local à la fois des agents et des fourmis, nous avons étendu le modèle de confiance {Vercouter and Jamont, 2012} au travail collaboratif des fourmis. Pour cela, dans un premier temps, nous avons intégré dans la règle de transition, les valeurs de confiance calculées par les agents afin d'éviter d'emprunter les régions du réseau contenant des agents en défiance avec leurs voisins. Puis, dans un second temps, nous avons identifié les informations qu'un comportement malveillant pouvait altérer, leurs impacts sur le comportement des fourmis et nous avons proposé des contre-mesures.

### 4.2.6 ≡MWAC : Simplifier l'observation

#### 4.2.6.1 Objectif

Les CCP se présentent souvent sous la forme d'assemblage de sous-systèmes qui peuvent être hétérogènes, ouverts, de grandes dimensions et possédant eux-mêmes une structure flexible. Il s'agit donc de système complexe au sens premier (*complexus* signifiant "tissé ensemble") dont l'observation est rendue difficile notamment par la diversité :

— des interactions possibles entre les différents composants élémentaires, les différentes organisations, les environnements qu'ils soient pris séparément ou ensemble ;
— des dynamiques spatio-temporelles résultant notamment de l'ajout et la suppression de composants, de buts, d'utilisateurs etc.

Comprendre un tel système nécessite d'identifier des caractéristiques communes à certaines parties du système : elles deviendront des abstractions. Il est donc nécessaire d'appréhender le système au niveau global (observer le tout à distance) et aux différents niveaux propres aux parties. Cette approche par strates et par niveaux d'abstraction permet de les rendre intelligibles en réduisant leur complexité (défi 6).

#### 4.2.6.2 Présentation succincte du modèle

Le modèle *Système Multi-Agent Réccursif (SMA-R)* {Hoang et al., 2011, 2012} issu de la thèse de H. T. T. Hoang, que nous avons co-encadré avec Michel Occello et Choukri-Bey Ben-Yellès, propose une approche multi-agent récursive pour permettre l'observation des systèmes complexes artificiels munis d'une organisation multi-échelle dans un but de supervision (visualisation, suivi et action sur la totalité ou certaines parties de ces systèmes). Un framework générique a été proposé, capable d'intégrer un SMA réel en agrégeant les entités du système pour construire des niveaux d'abstraction. Les couches d'abstraction sont implémentées sur les entités (agents) existantes. Cette section présente son instantiation au modèle MWAC qui a ainsi donné naissance au modèle ≡MWAC.

**Le modèle SMA-R {Hoang et al., 2012}**

*Démarche* Les CCP sont considérés au niveau d'observation le plus fin (niveau 0, c'est-à-dire le niveau réel). En regroupant les agents et les objets du niveau 0 sur certains critères comme la structure d'organisation, leurs relations ou leurs connaissances, on peut mettre en évidence des partitions ou des phénomènes du système. Ils correspondent à des réalités localisées liées à des critères précis d'observation (par exemple, "tous les capteurs d'une zone", "les services web rendant un service donné", "les robots participant à la tâche $X$", "les membres d'un réseau social possédant une compétence spécifique"...). Si on éloigne suffisamment l'observateur et que l'on observe un phénomène (composé d'individus) comme un tout, on peut alors passer à un autre niveau d'observation (niveau 1,...). À chaque niveau, chaque phénomène représente des individus agrégés du niveau inférieur donnant ainsi une nouvelle perception du système. La complexité du système est ainsi réduite et l'intelligibilité du système enrichie. Un niveau d'observation donne une vision abstraite du système. À travers les niveaux, l'intelligibilité et la complexité évoluent en sens opposés.

A chaque niveau les comportements sont en fait identiques. Des propriétés de transformation permettent de construire chaque niveau d'abstraction. Ces propriétés permettent de passer d'un niveau à l'autre à la fois en termes de comportement ou de structures. Elles sont identiques sur chacun des niveaux, elles constituent des propriétés récurrentes pour modéliser ces systèmes et leur dynamique {Occello, 2000}.

*La dynamique de la propriété récurrente* Le passage d'un niveau à l'autre se fait donc par application des transformations ($PA$, $PE$, $PI$ et $PO$) qui mettent en œuvre les règles qui fixent les modalités de passage d'un niveau vers l'autre.

L'appel dynamique de ces transformations permet d'augmenter l'adaptativité et la flexibilité des structures en cours de processus. Cette approche est particulièrement utile si le nombre de décompositions récursives est variable et n'est pas défini au préalable. Le développement d'un niveau de

récursion peut être créé dynamiquement d'après l'état courant du processus. Cela peut être réalisé par l'application dynamique de l'ensemble des transformations récurrentes définies pendant la phase de conception et qui sont dépendantes de l'application. Le nombre de niveaux maximum dépend du problème, c'est-à-dire du choix des règles $R$, de la méthode de construction des groupes.

*Le modèle multi-agent générique récursif* Un *agent applicatif* est un agent de l'application à laquelle on applique le modèle récursif, qu'on appellera aussi "agent du système" et qui peut être un agent physique comme c'est le cas dans notre contexte. Un *agent récursif* peut être *élémentaire* au niveau 0, associé à un agent applicatif, ou *composé* ou *abstrait*, associé à une abstraction du SMA applicatif des niveaux 1,2,...

Au niveau élémentaire les agents récursifs encapsulent ou interfacent les agents applicatifs et sont capables d'interagir par leur biais avec le système réel. Les agents composés se chargent de transmettre les messages à leurs agents fils et à leur agent parent ou interagissent dans leur société virtuelle. Ils héritent aussi des informations communes de leur sous-société. Autrement dit, ils jouent un rôle de représentant de tous les agents qui les composent. Le modèle d'agent récursif générique SMA-R est composé de cinq parties : les connaissances (K), les mécanismes d'observation (O), de composition (C), de réduction (R), d'interaction récursive (IR). Il est détaillé dans le tableau 4.2.

| P. | Contenu | Observations | P.R. |
|---|---|---|---|
| $K$ | Connaissances, Capacités pour la construction des abstractions, Fonctions dynamiques | Partie liée à l'application | $PA$, $PE$, $PO$ |
| | Niveau d'abstraction de l'agent, États de récursion | Connaissance du contexte récursif | |
| | Liste d'agents agrégés, Liste des objets agrégés | Connaissance du contexte récursif | |
| $O$ | Observation les états dans K pour appeler des fonctions $C$ ou $R$<br>Appel de Composition(), Appel de Réduction() | Partie générique : observation et Auto-évaluation des fonctions dynamiques<br>Partie liée à l'application : observation des états (dans la partie $K$) pour déclencher les $C$ (Composition) ou $R$ (Réduction) | |
| $C$ | Composition() | Partie générique : création d'un nouvel agent à partir d'une liste d'agents et d'objets agrégés du niveau inférieur | |
| $R$ | Réduction() | Partie générique : notification à l'agent père de sa disparition et suppression de lui-même. La profondeur de la structure arborescente se réduit. | |
| $IR$ | Envoi, réception, analyse, traitement, transformation, transfert des messages.<br>Mise à jour de $K$ | Partie générique : interaction du mécanisme récursif<br>Partie liée à l'application : interaction applicative | $PI$ |

TABLE 4.2 – Les parties de l'agent récursif générique

La partie $K$ est totalement dépendante de l'application. $R$ et le $C$ sont des mécanismes indépendants de l'application mais utilisent des règles de passage dépendant du problème. Le $O$ et le $IR$ ne sont liées que partiellement à l'application. L'ensemble de ces mécanismes permet de construire la récurrence sur l'organisation. Le détail des cinq parties est le suivant :

— Le $K$ (*Connaissances et capacités*) : les connaissances et les capacités dépendent de l'application. Elles décrivent les propriétés de transformation récurrentes $PA$, $PE$ et $PO$ utilisées pour induire les abstractions, des agents, des objets de l'environnement et de l'organisation.

Elles décrivent, en outre, des informations de contexte sur la dynamique récursive, calculées par des fonctions dynamiques comme :

- état de complexité : évalué par $F_{Complexité}$. Un agent est complexe s'il satisfait à une règle de regroupement et doit demander la création d'une abstraction,
- état de satisfaction : évalué par $F_{Satisfaction}$. Un agent est satisfait si son objectif est satisfait,
- état de stabilité de l'agent (lui-même) : $F_{SA}$. Un agent est stable si son contexte n'a pas changé depuis un certain temps (la *période de sureté*),
- état de stabilité de la société qu'il gère : évalué par $F_{SS}$, détermine si l'ensemble des agents représentés par l'agent courant sont stables.

— Le $O$ (*Observation*) : il se charge d'évaluer les états du $K$ à l'aide des fonctions dynamiques afin de déclencher si nécessaire les deux mécanismes *Composition* et *Réduction*,

— Le $C$ (*Composition*) : les niveaux d'abstraction sont construits grâce à ce mécanisme. Les agents récursifs dans le niveau inférieur créent le niveau supérieur s'ils satisfont la condition de composition. La mesure de la satisfaction est une implantation des règles données par le concepteur,

— Le $R$ (*Réduction*) : ce mécanisme réalise le processus de *Réduction* du niveau d'abstraction. Ce mécanisme est déclenché si un agent composé veut se détruire,

— Le $IR$ (*Interaction Récursive*) : cette fonction prend en charge d'envoyer, de recevoir, d'analyser, de transformer, de transférer les messages. Ici, le concepteur doit définir la propriété récurrente *PI*. On doit traiter deux types interactions :

- l'interaction système qui concerne les messages de construction et de gestion des couches,
- l'interaction du problème ou de l'application.

**Une architecture décentralisée** Dans le contexte des CCP et donc des SMA physiquement décentralisés, l'application du modèle SMA-R nécessite la proposition d'une architecture décentralisée.

*Influence du modèle OSI* Nous avons adopté les concepts du modèle OSI {ISO, 1989} pour traiter les messages dans cette architecture décentralisée. Le concept OSI repose sur la notion de couche de service. Chaque couche fournit un niveau d'abstraction, en cachant les détails des fonctions des couches inférieures et en offrant des services aux couches adjacentes. Les couches d'un même niveau ont leurs propres protocoles de communication qui servent à véhiculer leurs données (Protocol Data Unit (PDU)). Cependant les données ne sont réellement transférées d'un nœud à un autre que via la couche physique.

La généricité du modèle OSI vient du fait que les services entre les couches adjacentes sont exprimés en termes de primitives identiques pour chaque couche. Nous empruntons ce principe au modèle OSI. Dans notre modèle, la couche 0 est composée des agents élémentaires. Ces agents peuvent être physiquement reliés entre eux. Les couches supérieures utilisent les couches inférieures pour envoyer leurs données (il y a encapsulation). Les primitives sont les mêmes pour toutes les couches (propriété récursive). Le nombre de couches est fonction de la taille de l'application et selon la définition des propriétés récurrentes du concepteur. Les messages transmis entre les couches sont transformés pour s'adapter au protocole de la couche inférieure.

Pour que la mise en œuvre soit vraiment décentralisée, les agents récursifs sont attachés à chacun des agents applicatifs. À chaque agent applicatif est associé un agent élémentaire. Chaque fois qu'on construit un nouveau niveau, le nouvel agent composé s'emboîte sur l'agent voulant se composer. Tous les agents agrégés ayant le même agent parent auront le même agent récursif. Par exemple, dans la figure 4.7, nous appliquons le modèle récursif à cinq agents applicatifs. Au niveau 0, nous avons cinq agents (niveaux) élémentaires (cercles en trait continu) qui s'emboîtent directement sur les cinq agents applicatifs. Les mécanismes de *composition* et *réduction* construisent deux nouvelles couches : la première couche se compose de deux agents composés (cercles en pointillés), la deuxième est constituée d'un seul agent composé (cercles en pointillés).

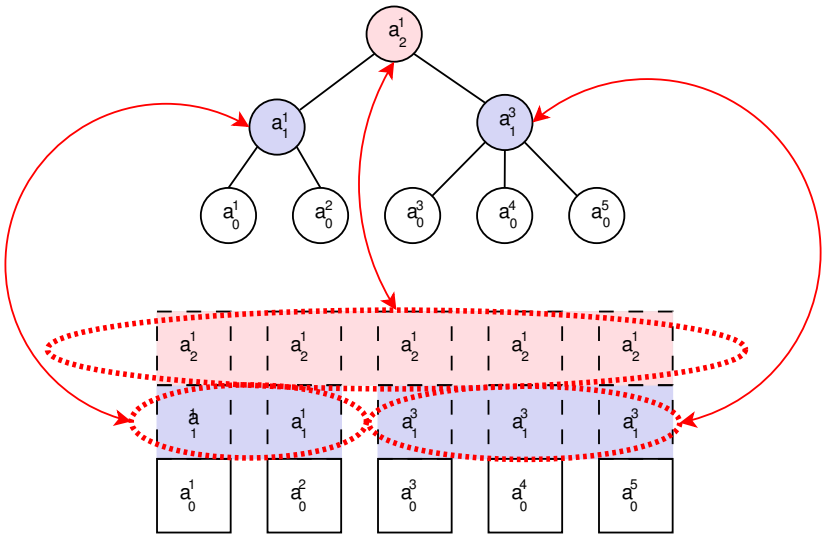


FIGURE 4.7 – Correspondance modèle / architecture

On remarque que chaque agent composé est implémenté par un morceau des données attachées à chaque agent de la couche inférieure. Sur la couche 1, l'agent composé $a_1^1$ est réparti sur les deux agents de la couche inférieure $a_1^0$ et $a_2^0$ (agrégés pour former l'agent $a_1^1$). Chaque agent élémentaire de la couche 0 est associé à un agent applicatif. Les couches ne peuvent communiquer qu'avec les couches adjacentes conformément au fonctionnement du modèle OSI.

*Description des couches récursives* Le diagramme de classe (figure 4.8) décrit les principes de l'architecture d'agent récursif. La classe principale est *RecursiveAgent* dont chaque instance est associée à un agent de l'application réelle (classe *ApplicativeAgent*) et qui possède deux associations réflexives : l'association *next* pointe vers l'agent père dans la couche supérieure et l'association *previous* pointe vers l'agent fils dans la couche inférieure. Cette classe contient les fonctions qui permettent de déclencher les *Composition* et *Réduction*. La classe *SystemMessage* contient les messages permettant le bon fonctionnement des mécanismes récursifs des agents.

Les informations relatives à la partie $K$ du modèle SMA-R sont l'identifiant (*id*), le niveau de récursion (*level*), l'agent applicatif, une liste d'agents agrégés etc. Le $O$ est implémenté dans la méthode *recursiveObservation()* qui sollicite la méthode *computeRecursiveState()* afin d'évaluer l'état de

l'agent (satisfaction, complexité, stabilité ...). En fonction de cet état, les méthodes *recursiveComposition* ou *recursiveReduction* seront appelées. Les parties $C$ et $R$ sont évidemment implémentés dans les méthodes *recursiveComposition* et *recursiveReduction*. Le $IR$ est interprété par les méthodes : *sendMessage(), receiveMessage(), processSystemMessage() et processApplicationMessage()*.

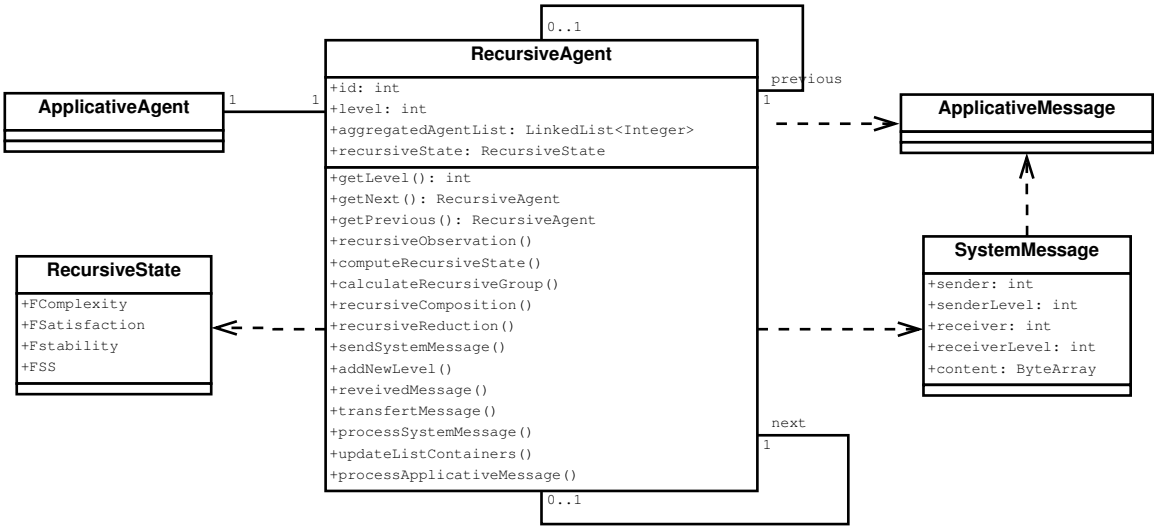



FIGURE 4.8 – Diagramme de classe du modèle multi-agent récursif

*Comportement des couches récursives* Il y a trois processus réalisés en parallèle :

1. Le premier calcule régulièrement les fonctions dynamiques et les états de récursion.
2. Le second observe les états de récursion, les fonctions dynamiques d'agents et déclenche les deux fonctions *réduction* et *composition*.

   Lorsque la fonction *Composition* est déclenchée :

   (a) L'agent crée son agent père (dans le niveau supérieur). Ensuite, l'agent lui-même s'ajoute à la liste des agents agrégés ;

   (b) L'agent envoie un message de l'invitation de composition *REQUEST_ COMPOSE* à d'autres agents dans son groupe ;

   (c) Lorsqu'un agent reçoit un message *REQUEST_ COMPOSE*, s'il accepte l'invitation de composition, il crée son agent père (dans la couche supérieure) avec les paramètres reçus dans le message *REQUEST_ COMPOSE* et renvoie un message *ACCEPT_ COMPOSE*, sinon il renvoie un message *REFUSE_ COMPOSE* ;

   (d) Si un agent reçoit une réponse *ACCEPT_ COMPOSE*, il ajoute alors l'expéditeur du message *ACCEPT_ COMPOSE* à la liste des agents agrégés et envoie à tous les agents de la liste des messages *UPDATE_ COMPOSE* ;

   (e) Lorsqu'un agent reçoit un message *UPDATE_ COMPOSE*, il doit mettre à jour l'information concernant la liste des agents agrégés et l'information de l'agent père pour que l'agent père de tous les agents agrégés soit identique.

Si la fonction *Réduction* est déclenchée :

(a) L'agent envoie un message *REQUEST_REDUCTION* à son agent père pour lui demander de se détruire ;

(b) L'agent père reçoit le message *REQUEST_REDUCTION*, s'il a un agent père, il lui envoie un message *RESIGNATION* pour l'avertir de sa disparition et se détruit lui même. S'il n'a pas d'agent père, il se détruit sans envoyer le message *RESIGNATION* ;

(c) L'agent père reçoit le message *RESIGNATION*, il supprime l'émetteur du message dans la liste d'agents agrégés.

3. Le troisième processus attend les messages provenant d'autres agents et les traite.

**Opérationnalisation dans le contexte de MWAC** À travers les parties $K$, $R$ et $C$, nous exprimons la propriété récurrente *PA*. L'organisation MWAC conditionne la règle de génération d'une abstraction de niveau supérieur. Une fois l'organisation MWAC stabilisée, les groupes MWAC formés permettent de construire un agent de niveau supérieur représentant le groupe.

Les cinq parties de l'agent récursif (tableau 4.2) appliquées au modèle MWAC sont détaillées comme suit :

— $(K)$ : contient les attributs et les fonctions de l'agent MWAC, en particulier les états de récursion. Les conditions de composition et réduction sont définies ici. Par exemple, la condition de composition est : *(recursiveState.*$F_{Stability}$*==1) and (getNext()==null) and (getRole()==roleREPRESENTATIVE) and (any neighbour's role == roleLINK))*.

— $(O)$ : si ces conditions de *composition* (ou *réduction*) sont satisfaites, le mécanisme *composition* (ou *réduction*) est lancé.

— $(C)$ : crée une nouvelle instance de la classe *RecursiveAgent* et l'assigne à *next* pour construire une nouvelle couche. La propriété récurrente $PA$ définit la liste *aggregatedAgentList*.

— $(R)$ : avertit de sa disparition l'agent père et se détruit.

— $(IR)$ : gère deux types d'interactions :

  — Interaction système : les messages REQUEST_COMPOSE, ACCEPT_COMPOSE, REFUSE_COMPOSE, UPDATE_COMPOSE,... ;

  — Interaction applicative : tous les messages de MWAC.

  La partie *IR* nous permet d'écrire la propriété récurrente *PI* :

  — Pour les messages MWAC dont le destinataire est "tout le monde" : le message va être distribué à tous les agents élémentaires qui se situent sur les feuilles dont la racine est l'agent émetteur du message. Les agents élémentaires vont envoyer le message de l'émetteur.

  — Pour les messages exprimant une requête, il est nécessaire d'expliciter la méthode de fusion des informations. Par exemple, pour la requête "demander une mesure de température à l'environnement", la température mesurée d'un agent composé est définie par la moyenne arithmétique des températures captées des agents dans la liste *aggregatedAgentList*.

La figure 4.9 monte un même CCP vu à deux niveaux d'abstraction différents : le niveau dit élémentaire (en haut à gauche de la figure) et les autres niveaux d'abstraction. Au dernier niveau (niveau 4) il n'existe qu'un seul groupe.

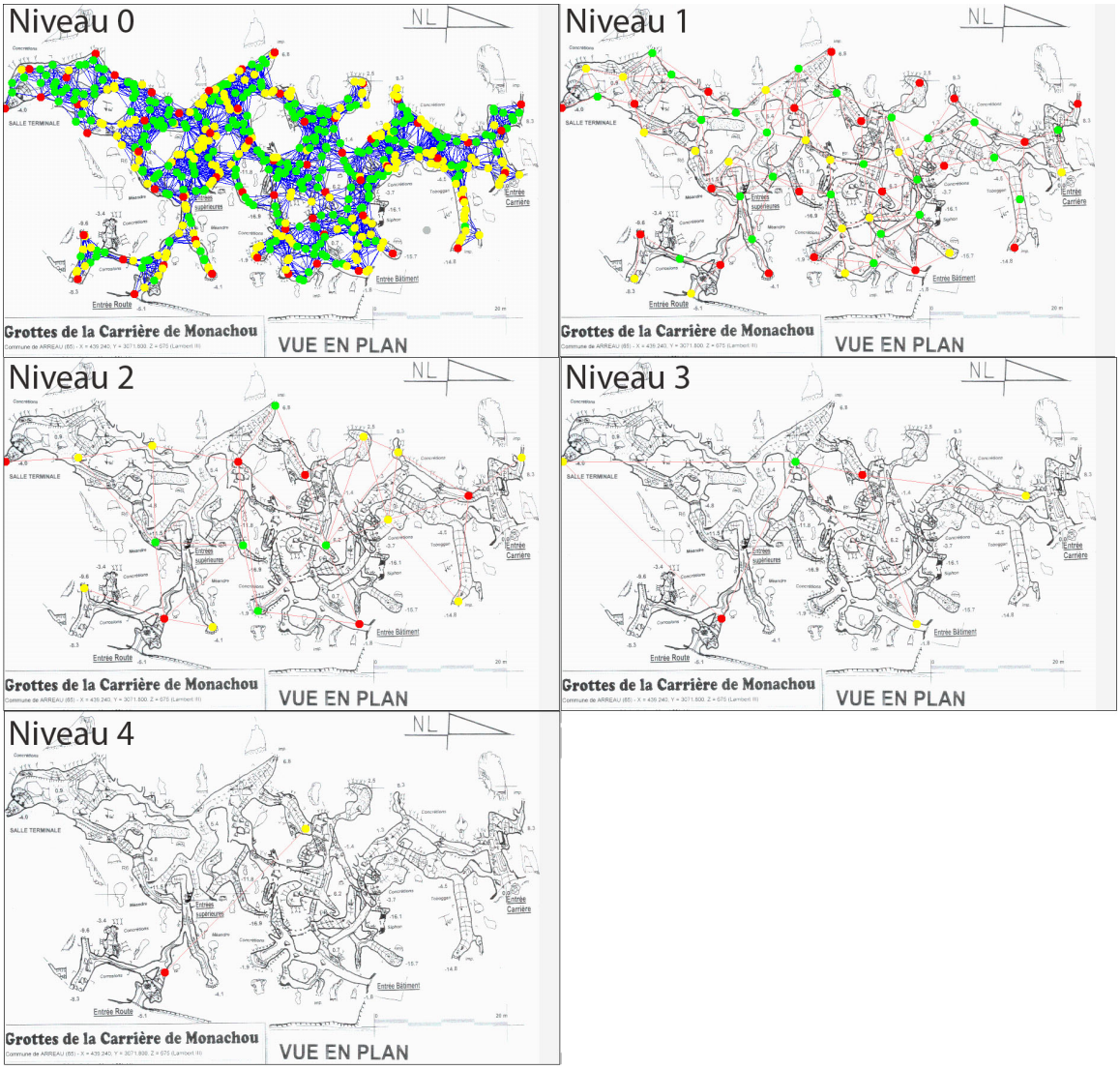



FIGURE 4.9 – Un CCP vu à ses différents niveaux d'abstraction

#### 4.2.6.3 Conclusion

Par l'utilisation d'un modèle générique de SMA récursif nous avons permis de simplifier l'observation d'un CCP implantant MWAC.

Cette implantation est compatible avec nombre des contraintes non fonctionnelles liées aux CCP. Elle n'est pas centralisée sur un nœud ou une infrastructure dédiée qui aurait d'importantes ressources lui permettant d'établir un modèle complet du système. Notre contribution peut être considérée comme un middleware décentralisé. Le framework empile sur chaque agent et pour chaque niveau d'abstraction une couche possédant des informations organisationnelles locales et les primitives des mécanismes récursifs. Le framework encapsule les agents applicatifs et peut donc être totalement dissocié de l'application.

La limitation d'une telle approche dans le contexte des CCP est liée à l'importance que prennent

les échanges de message :

— l'utilisation du modèle OSI augmente le volume quasiment linéairement avec le nombre de niveaux. En effet, quand un message est émis à un niveau $n$, le message applicatif sera encapsulé $n$ fois ;

— l'implantation totalement décentralisée nécessite qu'un événement généré à un niveau $n$ soit propagé à l'intégralité des agents de niveau $n-1$.

Le détail des mécanismes et l'évaluation du modèle est proposée dans le papier référence du modèle {Hoang et al., 2012}.

## 4.3 Des modèles pour les collectifs à environnements changeants

### 4.3.1 Motivations

Ce travail a été suscité par une collaboration avec l'équipe de Félix Ramos au CINVESTAV. Il fait l'objet de la thèse de Francisco Cervantès que j'ai co-encadré avec Félix Ramos et Michel Occello. L'objectif est d'assurer le maintien de services à des patients qui se déplacent fréquemment dans différents environnements (des parcours du type maison médicalisée → ambulance → hôpital). Nous nous sommes donc particulièrement intéressés aux aspects liés à la sélection de services et au maintien des compositions de services faisant sens pour l'utilisateur en environnement changeant. Ne pas travailler avec des environnements clos implique l'absence d'une infrastructure commune de communication et rend difficile la centralisation du processus de composition.

L'informatique ambiante est centrée sur l'utilisateur humain : elle a pour but d'exploiter les objets communicants de son environnement pour répondre à ses attentes. Pour cela, elle repose sur une forte utilisation des communications sans fil et suppose souvent l'existence d'une infrastructure pour mettre en relation les objets connectés et pour mettre en œuvre leur coordination. Cette infrastructure est de plus souvent connectée à Internet.

De nombreux travaux se focalisent sur la composition des services fournis par ces objets. Les architectures orientées service (SOA) se sont imposées car elles proposent un faible couplage des services impliqués dans les fonctionnalités. Cependant il subsiste de nombreuses limitations lorsque l'on s'intéresse aux services ayant des dépendances avec des grandeurs physiques. Supposons par exemple que l'agent assistant d'un utilisateur travaille à assurer son confort thermique. Si cet utilisateur était dans son appartement et qu'il prend son véhicule, le changement d'environnement rend obsolète l'utilisation du service chauffage de son domicile.

Parmi les défis qui doivent encore être relevés {Issarny et al., 2011} on peut citer le fréquent problème du passage à l'échelle (défi 6), l'interopérabilité des objets (défis 8 et 9) notamment d'un point de vue sémantique, la mobilité qui renforce la dépendance au contexte (défi 11) et la gestion de la sécurité au sens large (défis 5 et 12).

### 4.3.2 Description succinte

Les SOA prônent le découplage entre consommateurs et fournisseurs de services en proposant des standards pour réduire la dépendance notamment des services. Fortement inspiré du découplage entre classe dans les approches orientés objets, il vise l'interchangeabilité des services.

Le découplage rencontré dans les SMA va plus loin car les composants élémentaires que sont les agents ont une réelle autonomie de décision. C'est ce qui explique l'occurrence de nombreux travaux {Richards et al.; Maamar et al., 2005; Morais, 2009; Gutiérrez-García et al., 2012} où les agents agissent en tant que médiateurs dans le modèle fonctionnel des services et dans lesquels les services peuvent être intégrés dans les agents sous la forme de capacités.

**Une approche multi-agent inspirée par les "écosystèmes"** Nous avons approché ces collectifs cyber-physiques constitués d'objets connectés comme des "écosystèmes" afin d'obtenir une organisation flexible et une adaptation forte à l'environnement. Si la notion d'adaptation des espèces à l'environnement est un aspect qui faisait écho à nos objectifs, Francisco a été séduit par cette métaphore utilisée dans de plusieurs domaines pour décrire des processus collectifs {Chesbrough, 2006; Fréry et al., 2012}.

Cette métaphore nous a amenés à proposer une décomposition en 3 couches du processus qui consiste à délivrer et maintenir des services aux utilisateurs en environnement changeant (voir figure 4.10). La première de ces couches est constituée de l'ensemble des objets communicants hébergés par les éventuelles infrastructures sous-jacentes. Ces objets qui proposent différents services sont modélisés par des agents en respectant la décomposition inhérente aux écosystèmes. Par exemple, les espèces regroupent des membres de l'écosystème qui partagent des caractéristiques, des compétences, des besoins ou bien sûr des buts. Les membres interagissent en respectant des normes sociales pour répondre aux attentes des utilisateurs.

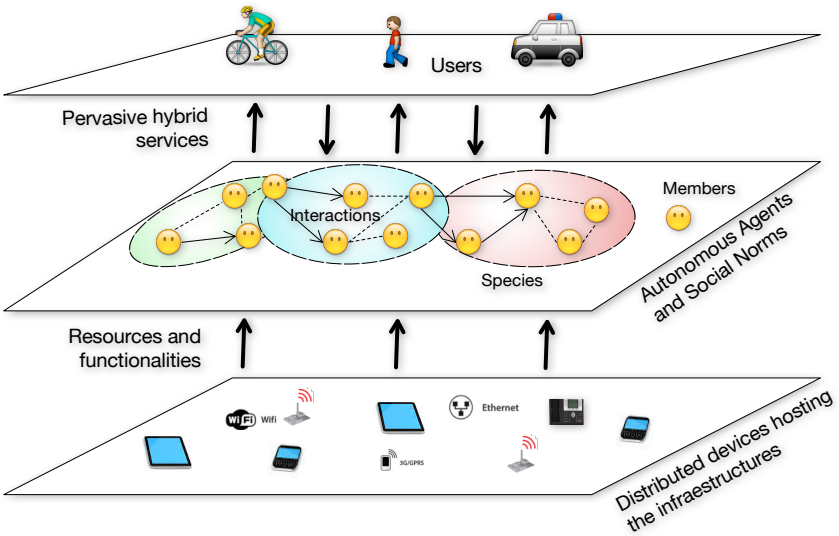



FIGURE 4.10 – Illustration de notre approche basée "écosystèmes"

**Obligations sociales** La production de services pour les utilisateurs nécessite la coopération de membres d'une espèce et d'espèces différentes. Les interactions entre ces agents respectent des règles que fixent les protocoles d'interactions. Dans ce contexte, les approches mentalistiques s'intéressent à l'effet du message sur l'état mental du destinataire. Ces approches ne sont adaptées ni aux environnements ouverts ni à la prise de décision concertée {Singh, 1998; Morge, 2005}. Des extensions sont proposées pour inclure des normes sociales publiques {Singh, 2000; Gaudou et al., 2006; Gutiérrez, 2009} mais elles ne sont pas adaptées aux environnements ouverts. Pour lever ce verrou, dans le prolongement d'autres travaux de l'équipe de Félix Ramos, la notion d'obligation sociale est proposée. Nous avons notamment défini un cycle de vie (voir figure 4.11) des obligations pour adapter leur utilisation aux contraintes des environnements ouverts et dynamiques. L'état d'adaptation permet de trouver dans le collectif un autre membre capable d'assumer l'obligation. Plus loin, un historique des états d'une obligation peut permettre l'identification de la criticité d'une obligation.

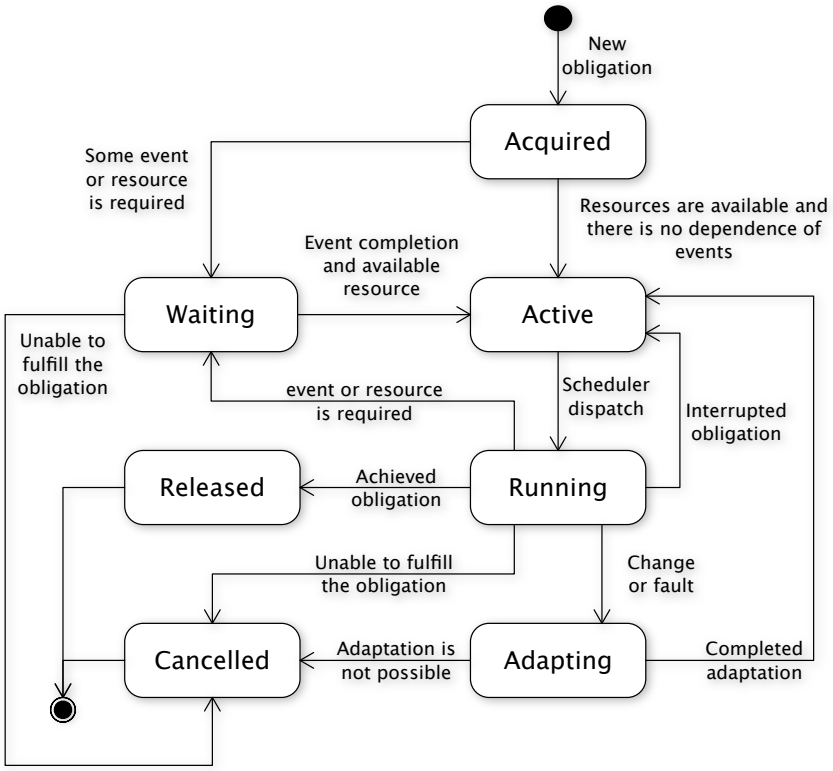



FIGURE 4.11 – Le cycle de vie des obligations

**La composition et l'adaptation de services ambiants.** Nous étendons le modèle proposé par Karmouch and Nayak {2012} pour faire face aux effets de la dynamicité de l'environnement, de la mobilité des nœuds et des communications intermittentes. Ainsi le problème est formulé comme un problème de satisfaction de contraintes distribuées et sa résolution avec un algorithme à retour de traces. Notre extension permet notamment de ne pas remettre en cause toute la composition quand une erreur intervient pendant son exécution. De plus, un plus grand nombre de critères sont pris en

compte lors de la sélection des services comme la qualité des transmissions.

### 4.3.3 Discussion

Le détail de ces travaux développés dans la thèse de Francisco Cervantès est en parti publié dans {Cervantes et al., 2014}.

Les résultats des premières expérimentations ont été obtenus uniquement par simulation. Ils valident pour différentes applications le processus de création et d'adaptation des compositions de services en étudiant notamment l'impact de la densité des services. Un déploiement réel demandera des adaptations comme par exemple l'exploration multi-saut des infrastructures. Actuellement nous exploitons les services dans le voisinage relativement immédiat de l'utilisateur (les objets des infrastructures voisines). Étant donné la densité et la dispersion des services escomptées dans un environnement réel, il est nécessaire de prévoir des explorations multi-sauts guidées pour limiter l'impact sur la consommation d'énergie.

## 4.4 Conclusion

Nous avons présenté dans ce chapitre les caractéristiques des CCP et les défis que devaient relever les concepteurs afin de permettre d'exploiter les modèles multi-agents de coopération.

### 4.4.1 Des modèles d'agents et de collectifs opérationnels

Un CCP doit répondre aux exigences des utilisateurs artificiels et humains en respectant des contraintes non fonctionnelles qui expriment la qualité du service attendu. Nous considérons le CCP comme un SMA composé d'agents disposant :

— de compétences souvent insuffisantes pour lui permettre de répondre seul à l'exigence,
— de ressources limitées qu'il doit gérer lui-même et éventuellement partager avec d'autres,
— de capacités de communication non fiables.

Les approches multi-agents permettent de trouver une réponse aux exigences qui ne se situent pas dans la sphère individuelle de l'agent mais dans sa sphère sociale. Elles permettent donc aux agents d'outrepasser leurs limites en proposant une réponse collective. Les architectures d'agents vont alors guider l'intégration des différentes facettes de la décomposition multi-agent dans les agents. Les architectures permettent donc, en un sens, de lever le critère d'extériorité du phénomène collectif.

Nous avons proposé deux architectures d'agents compatibles avec les défis imposés par la nature physique des systèmes qui nous intéressent :

— *eASTRO* est une architecture d'agent embarqué permettant une mise en œuvre de CCP modélisé par des *SMA* sur des architectures à base de microcontrôleurs. Cette architecture que nous utilisons le plus lors de déploiements réels relève les défis aux limitations des ressources de calcul et de stockage.
— *Avatar* nous permet d'étendre les connaissances et le comportement d'objets physiques dont le raisonnement est contraint par des limitations matérielles importantes. Cet enrichissement provient des services Web que l'avatar découvre et par les connaissances qu'il utilise dans

le Web des Données. Ce modèle est implanté notamment par la société Génération Robots spécialisée dans la robotique de service.

Nous proposons aussi des modèles centrés sur les facettes Interaction et Organisation d'un CCP. Ils peuvent être vus comme :

— des modèles pour la gestion adaptative des communications comme nous l'avons fait dans le contexte de l'instrumentation d'environnements naturels (projet FITT EnvSys {Jamont et al., 2010}, projet Tassili Hamani et al. {2011}), la prédiction de crues en Chine (travail mené conjointement avec l'équipe du professeur Zhongzhi Shi, Institute of Computing Technology, Chinese Academy of Sciences {Luo et al., 2007}) ou d'autres travaux au Mexique qui sont inspirés des nôtres {Olascuaga-Cabrera et al., 2011, 2012} ;

— des processus organisationnels utilisables dans des divers contextes applicatifs comme l'a montré par exemple un travail interne à l'équipe Cosy (mené par Annabelle Mercier) sur la recherche de compétences dans le réseau social d'une entreprise {Mercier et al., 2012} ;

— des modèles facilitant l'utilisation d'autres modèles multi-agents en permettant l'abstraction de certaines contraintes physiques (liées aux défis que nous avons précédemment mentionnés).

Enfin, nous avons proposé des modèles pour le collectif centrés sur les facettes Interaction et Organisation dans lesquels l'Environnement était changeant.

### 4.4.2 De nouveaux défis à relever

**Du contraint au fortement contraint** Dans les projets industriels qui nous ont permis d'éprouver nos approches et nos modèles, les contraintes issues des différents domaines applicatifs ont permis l'implantation des agents sur des plateformes à base de micro-contrôleurs. Cependant des contraintes fortes sur, par exemple la consommation d'énergie ou le dégagement thermique du système, comme c'est le cas dans l'aéronautique, ne permettraient pas de tels déploiements. Aussi construire des architectures d'agent répondant à des contraintes encore plus fortes que celles que nous avons traitées jusqu'à présent soulève de nouveaux défis. Par exemple, s'il est commode de supprimer la capacité d'un agent à accéder aux descriptions de ses actions afin de minimiser l'empreinte mémoire de son architecture, cela nuit à la préservation de ses capacités d'introspection et donc d'adaptation.

**Des petits aux grands mondes** Dans toutes nos applications, les agents appartiennent à un même CCP. Ils ne coopèrent que dans un cadre relativement bien délimité et n'interagissent donc qu'avec des agents conçus pour coopérer avec eux [10]. Même les applications envisagées dans le cadre du projet ANR ASAWoO sont en réalité des projets "intraweb des objets" plutôt que WoT.
Si on relaxe l'hypothèse des petits mondes, lorsqu'un agent est sollicité par des agents d'autres SMA que le sien, on peut lui imaginer plusieurs états d'esprit. En s'inscrivant dans le prolongement des travaux multi-agents, l'agent est vu comme une entité coopérante par essence. L'occurrence d'objectifs incompatibles entre les agents de SMA différents serait alors vu comme un conflit que l'on va résoudre en négociant. Une autre piste consisterait à remettre en cause cette attitude purement coopérative et rendre l'agent plus précautionneux : avant de répondre à une sollicitation externe, il mesurera autant que possible les implications pour son propre système.

10. Nous ne considérons pas ici l'interaction avec un agent malveillant comme une interaction avec un agent extérieur au SMA.

# Chapitre 5

# Un outil pour la mise au point de collectifs cyber-physiques

De plus en plus de systèmes intelligents embarqués dans le monde réel se composent de nombreux dispositifs inter-connectés qui interagissent ensemble. Enfouis, ils sont de petites dimensions, peu coûteux et donc aussi simples que possible. La conception de ces systèmes complexes nécessite de prendre en compte des contraintes non fonctionnelles fortes comme la limitation de la consommation d'énergie.

Deux grands types d'approches sont utilisées lors de leur conception :

1. les approches centralisées qui sont les plus répandues dans l'industrie, qui se basent sur un modèle global de l'environnement (i.e. le système à contrôler). Le processus décisionnel du système utilise ce modèle pour décider et agir.
2. les approches décentralisées, que nous défendons, qui voient le système comme un CCP c'est-à-dire un ensemble d'entités logicielles et matérielles coopérantes bien qu'au comportement autonome et qui raisonnent avec des descriptions partielles de leur environnement.

Dans le premier cas, l'obtention d'un modèle global réaliste peut être coûteux en terme de calcul et en terme de transport d'information. Dans le second cas, il est plus difficile de prouver et donc de garantir des propriétés comme la stabilité. De plus, l'éventuel nombre de messages échangés par les entités distribuées pour permettre la coopération peut devenir important.

## 5.1 Motivation

Une solution naïve consiste à mettre en œuvre le système SMA qui modélise le CCP directement sur les plateformes cibles. Cette solution, envisageable pour des systèmes réactifs, compliquera la conception et le développement de la majorité des applications notamment lorsque la complexité est importante (défi 6). Une des raisons pour laquelle cette démarche est en effet coûteuse, que ce soit financièrement ou en temps de développement, est que les modèles développés sont souvent spécifiques au couple que forment l'application et la plateforme matérielle utilisée. De plus, les

difficultés liées à l'embarquement du SMA s'ajoutent aux difficultés purement fonctionnelles. Le caractère embarqué est assimilable à un ensemble de contraintes en terme de ressources disponibles (limitation mémoire, faible puissance des CPU, partage du temps CPU etc.) ou d'autres contraintes non fonctionnelles comme le dégagement thermique, la consommation énergétique etc. Enfin, avec une telle approche, il est difficile de tester le SMA embarqué car aux éventuelles erreurs logicielles peuvent s'ajouter les erreurs matérielles. Ces erreurs peuvent notamment résulter de la mauvaise connaissance du matériel par les spécialistes du logiciel. Le problème des erreurs intermittentes est d'ailleurs particulièrement difficile à traiter à ce niveau.

Aussi, une alternative très populaire consiste à reproduire l'environnement du système et du SMA, à concevoir dans un environnement virtuel via un simulateur. Quand les résultats de la simulation sont en accord avec les performances attendues du système en cours de conception, on procède à son embarquement. Les inconvénients de cette méthode sont d'une part, la pertinence des résultats qui dépend de la qualité des modèles utilisés dans l'outil de simulation et d'autre part, le fait qu'à ce niveau, l'application simulée est totalement découplée des contraintes matérielles. La simulation terminée, un effort important de développement est nécessaire dans le sens où il faut à nouveau développer mais, cette fois-ci, le code embarqué. Lors de la création du code "embarqué" il est possible que des déviations apparaissent par rapport au système simulé originellement. Ces déviations proviennent de la dégradation (voulue ou non) des modèles pour s'adapter aux ressources disponibles et aux autres contraintes.

L'origine de nos travaux, dans ce contexte, est un constat issu de notre propre expérience : la simulation logicielle des SMA embarqués ne suffit pas ! En effet, aux difficultés de la conception traditionnelle de SMA logiciels s'ajoutent les problèmes de déploiement sur des plateformes physiques, le test du SMA embarqué et son débogage. De plus, le passage à l'échelle est aussi un problème très contraignant dans notre contexte (nécessité d'avoir un nombre important d'équipements à disposition).

Notre conviction a été qu'un nouveau type d'outil de simulation pour développer des SMA embarqués était nécessaire afin de permettre au concepteur d'explorer les différentes stratégies locales de contrôle et de mesurer leur impact sur le comportement global du système.

Il doit prendre en charge une variété importante de modèles provenant de différents domaines. Ces modèles sont principalement :

- des modèles d'organisation,
- des modèles d'interaction,
- des modèles des parties matérielles qui constituent l'agent (batteries, effecteurs, capteurs),
- des modèles de l'environnement physique liés à l'application, qui suivant leurs criticités pour l'application, peuvent comprendre des modèles de propagation des ondes, de dissipation thermique, etc.
- des modèles de comportement d'utilisateurs.

La précision de ces modèles utilisés dans la simulation a un impact significatif sur la qualité de l'ensemble de la solution qui sera déployée dans le monde réel.

En 2002, nous avions développé avec Michel Occello et André Lagrèze le simulateur SimEnvSys mis en œuvre dans le cadre du projet FITT ENVSYS. Rapidement, durant ma thèse, il m'a fallu in-

tégrer des modèles réalistes d'environnement (notamment ceux développés par Olivier Schira {2003} ( *DEA Image, Signal et Parole* effectuant son stage de recherche dans notre équipe) qui recréait les conditions de propagation d'ondes dans le contexte des réseaux hydrographiques souterrains). La volonté de permettre l'usage de cet outil dans d'autres applications, notamment dans le cadre du projet FITT PALETTE, a donné naissance à l'outil MultiAgent Software and Hardware simulator (MASH).

De nombreux étudiants de Master que nous avons accueillis ont travaillé avec l'outil, que ce soit directement pour son développement (Łukasz Nocuń, Sylwia Gasior, Zbyszek Królikowski) ou pour valider son usage dans des contextes applicatifs réels (Farid Jebahi, Paul Gery, Emmanuel Jez).

La section 5.2 introduit l'architecture de l'outil MASH. Dans la section 5.3, nous nous concentrons sur les principales caractéristiques de notre outil : la simulation de société mixtes logicielles/matérielles et l'utilisation de modèles physiques réalistes. Pour chacune de ces caractéristiques clés des travaux connexes sont cités.

## 5.2 Présentation succincte de MASH

### 5.2.1 Définitions préliminaires

Nous proposons ci-dessous une terminologie regroupant nos propres définitions des différentes notions utilisées dans le contexte de MASH. Certaines ont été introduites dans {Jamont et al., 2013}.

**Les agents** Les types d'agents rencontrés dans MASH qui interagissent dans un CCP sont exposés dans l'ontologie 5.1 :

1. Comme dans tout outil de simulation multi-agent, MASH manipule des *agents logiciels*.

   **Définition: Agent logiciel**
   Les agents logiciels sont des agents qui perçoivent et agissent dans l'environnement logiciel MASH.

2. Les *agents embarqués* dans le monde réel peuvent percevoir et agir sur l'environnement physique. Ils sont souvent constitués d'une partie logicielle et d'une partie matérielle.

   **Définition: Agent embarqué**
   Un agent embarqué est un système embarqué autonome qui dispose de sa propre unité de calcul, indépendante de l'outil MASH.

3. Un *agent virtuel* est un agent logiciel utilisé pour simuler le comportement d'un agent embarqué.

   **Définition: Agent virtuel**
   Un agent virtuel simule un agent embarqué c'est-à-dire que MASH reproduit son comportement avec ses propres ressources de calcul.

4. Un *agent proxy* est associé à un agent embarqué afin de permettre son intégration dans MASH et ainsi d'interagir avec les autres agents (logiciels ou embarqués).

   Considérons un agent comme un 4-tuple $O =< G, B, K, S >$, où $G = \{G_1, G_2, ..., G_n\}$ est un ensemble de buts (ce que l'agent veut faire), $K = \{K_1, K_2, ..., K_n\}$ ses connaissances c'est-à-dire un ensemble d'abstractions de son environnement et sur ses composants, $S = \{S_1, S_2, ..., S_n\}$ l'ensemble de services qu'il peut fournir et $B = \{B_1, B_2, ..., B_n\}$ son comportement c'est-à-dire les actions qu'il entreprend à partir de ses perceptions.

   > **Définition: Agent proxy**
   > Un agent proxy est un artefact de MASH qui lui permet d'interagir avec l'agent embarqué que ce soit pour accéder à son état, exposer ses connaissances et compétences, etc.

   Dans notre contexte, l'agent proxy est une projection de l'agent embarqué dans la société mixte logicielle/matérielle maintenue par MASH. C'est donc un intermédiaire entre l'agent embarqué et l'outil de simulation qui traite les requêtes de MASH. On peut définir cet agent proxy comme un 2-tuple $P =< Kp, Sp >$ où $Kp \subseteq K$ et $Sp \subseteq S$.

5. Un *avatar* représente et étend le comportement d'un agent embarqué dans MASH. Ce n'est pas un simple agent proxy : c'est une réelle entité autonome qui a son propre 4-tuple $A =< Ga, Ba, Ka, Sa >$ où $G \subset Ga$, $B \subset Ba$, $K \subset Ka$ et $S \subset S_a$. L'augmentation des connaissances et des comportements provient des informations ou des compétences des autres agents mais pourrait tout aussi bien provenir du Web comme nous le faisons dans l'ANR ASAWoO {Jamont et al., 2014a; Mrissa et al., 2015}.

   > **Définition: Avatar**
   > Un avatar est un agent logiciel qui étend le comportement d'un objet physique (système embarqué ou objet tagué RFID) à partir de connaissances et comportements accessibles via la plateforme MASH.

   L'objet physique et son avatar forment l'objet cyber-physique par excellence.

**L'environnement virtuel** Nous entendons par *environnement virtuel* l'environnement du SMA embarqué dans lequel les grandeurs physiques (températures, pressions, vitesses...) sont soit mesurées soit estimées à partir de modèles physiques. Les valeurs des paramètres de l'environnement physique sont acquises par les agents à l'aide de leurs capteurs (réels ou virtuels) et sont modifiées par leurs effecteurs (réels ou virtuels).

> **Définition: L'environnement virtuel**
> Environnement dans lequel les différents agents évoluent et dont les paramètres sont maintenus par MASH en utilisant des capteurs/effecteurs physiques (ceux qui sont plongés dans le monde réel) ou virtuels (ceux qui agissent sur les modèles virtuels d'environnements physiques).

**La société virtuelle** Nous appelons *société virtuelle* l'ensemble des agents logiciels du SMA. Un agent embarqué n'est alors pris en considération qu'au travers de son proxy ou de son avatar qui sont bel et bien des agents logiciels.

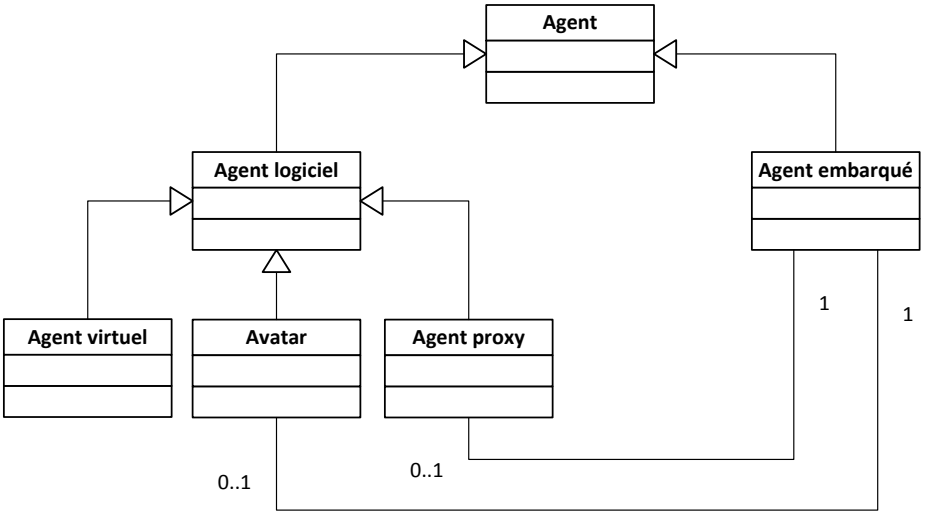



FIGURE 5.1 – Différents types d'agents évoluant dans MASH

**DÉFINITION: Société virtuelle**

Ensemble des agents logiciels mis en jeu dans MASH

**Les simulations** MASH permet plusieurs types de simulation de SMA embarqués :

1. La simulation dite *purement logicielle* dans laquelle chaque agent embarqué est simulé (utilisation d'agents virtuels). A ce niveau, le simulateur permet d'évaluer l'architecture logicielle des agents et de vérifier que la fonctionnalité globale du système est conforme à ce qu'on attendait d'elle.

   **DÉFINITION: Simulation purement logicielle**

   Simulation dans laquelle ne sont impliqués que des agents logiciels, les agents embarqués étant simulés.

2. La simulation précédente permet de valider la pertinence du SMA qui n'est à ce stade que purement virtuelle. La simulation dite *hybride logicielle/matérielle* va permettre d'inclure des agents embarqués dans la simulation. Elle permet notamment de tester les agents embarqués et de mesurer l'impact d'éventuelles déviations sur la fonctionnalité globale du système.

   **DÉFINITION: Simulation hybride logicielle/matérielle**

   Simulation d'un SMA contenant des agents logiciels et des agents embarqués (qui peuvent éventuellement interagir).

3. Les deux simulations précédentes ont permis d'obtenir un SMA conforme aux spécifications initiales. La simulation matérielle permet de tester le SMA dans un environnement agressif. Dans cette simulation, il n'y a aucun agent virtuel : le processus décisionnel des agents embarqués est donc exécuté sur leurs propres plateformes matérielles. Cependant, les agents interagissent

entre eux via l'environnement simulé par l'outil MASH.

> **Définition: Simulation purement matérielle**
> Simulation dans laquelle aucun agent embarqué n'est simulé.

**Les protocoles** Un agent embarqué implémente ses propres protocoles applicatifs : un protocole de communication pour transmettre des messages aux autres agents, un protocole d'introduction pour l'introduire dans une organisation, un protocole de négociation pour requérir des compétences etc. La mise en relation d'un agent embarqué et de MASH nécessite aussi des échanges de messages. Ces messages permettent par exemple à l'agent embarqué d'exposer dans MASH son état mental, de signaler des évènements le concernant (niveau d'énergie qui passe sous un seuil, saturation de la mémoire disponible etc.), lui permettant d'accéder à l'environnement virtuel etc. Nous avons défini pour cela un protocole d'échange que nous appelons *protocole système MASH*. Ce protocole est compact pour perturber aussi peu que possible le comportement de l'agent.

> **Définition: Protocole système MASH**
> Protocole implanté par un agent embarqué qui souhaite pouvoir accéder à des fonctionnalités spécifiques de MASH : notification d'évènements, envoyer un message à un agent logiciel etc.

La figure 5.2 précise le format de ce protocole et l'illustre avec trois exemples d'utilisation issus de son utilisation dans le cadre du projet Kurasu avec le CEA-LITEN.

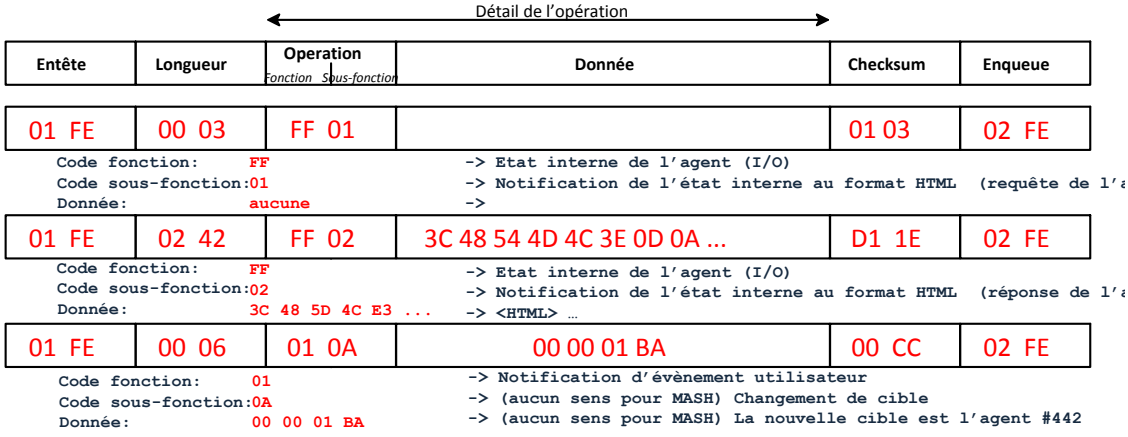



FIGURE 5.2 – Format du protocole système MASH

**Les instructions** Dans MASH, nous appelons *instruction* un appel de méthode sur un agent (ou un objet) à une date déterminée `(date,<object_identifier>.<method_name>(<param1>,...))`. Un *scénario* est un ensemble d'instructions.

Il est possible qu'une instruction agisse comme une violation de l'autonomie décisionnelle de l'agent : cela fait sens dans une phase de débogage des agents ou d'un collectif.

### 5.2.2 Cycle de vie associé à l'outil

L'objectif de MASH est de couvrir au mieux le cycle de simulation et de déploiement d'un CCP vu comme un SMA embarqué. Afin de montrer sa généricité et qu'il peut être utilisé indépendamment de la méthode DIAMOND, nous dissocions sa présentation de notre méthode. La figure 5.3 illustre cette démarche et positionne la simulation hybride logicielle/matérielle que permet notre outil dans la démarche de développement d'un SMA embarqué et par rapport à la simulation logicielle traditionnelle des SMA.

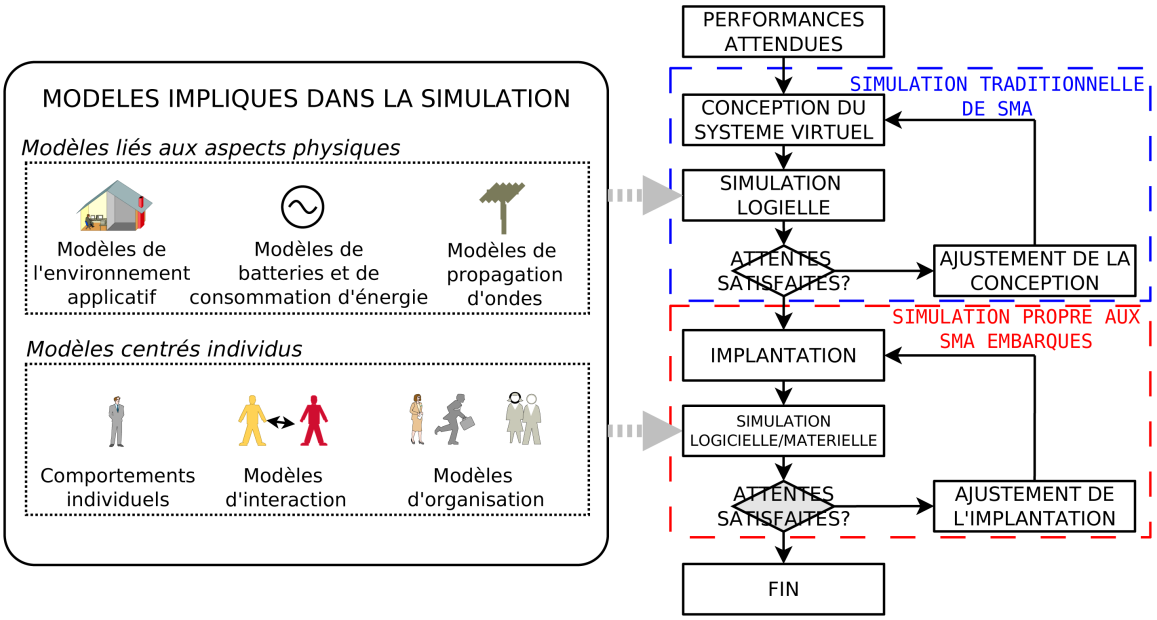



FIGURE 5.3 – Utilisation de MASH pour concevoir et déployer un CCP

À partir des besoins applicatifs et notamment des performances attendues, un prototype virtuel du SMA est conçu. Le concepteur simule alors le SMA et l'ajuste afin que les résultats mesurés soient en phase avec les besoins exprimés. Cette première phase présente de nombreux avantages, elle permet notamment :

— d'adresser la complexité de l'application à concevoir (défi 6) en la séparant de la complexité technique d'implantation sur des plateformes physiques et avant même que le matériel ne soit disponible ;

— de valider le comportement du système global et des entités qui le composent sur de nombreux scénarios dans un temps non comparable avec celui demandé dans le cas d'un réel déploiement. L'utilisation de scénarios adaptés permet ainsi au concepteur de focaliser certains comportement du système : la gestion de la sécurité (défis 5) ou la capacité du système à maintenir son intégrité fonctionnelle (défi 12) en utilisant des scénarios liés aux modes de défaillance , sa capacités à gérer son ouverture (défis 8, 9, 10) en utilisant des scénarios produisant des dynamiques exagérées d'ajout/suppression de sous-systèmes, la gestion de la mobilité (11) etc.

— d'observer des critères internes aux architectures d'agent beaucoup plus facilement que dans un contexte matériel.

L'utilisation de modèles réalistes de batterie et de consommation d'énergie permettra au concepteur d'avoir une idée de la bonne prise en compte, dans les cycles de décisions des agents, des contraintes liées à son autonomie en énergie (défi 4).

Une fois cette étape terminée, le concepteur peut s'intéresser à l'implantation du SMA sur une plateforme spécifique (cas du déploiement sur une plateforme commerciale) ou construire la partie matérielle de l'agent. Le simulateur va accompagner l'embarquement du SMA en assurant une meilleure gestion des risques qu'avec des outils traditionnels de simulation du domaine parce qu'il permet la simulation conjointe logicielle/matérielle. Le risque majeur est l'introduction de déviations dans le fonctionnement global du SMA provenant de la dégradation des modèles originels mis en œuvre dans le SMA prototypé virtuellement pour prendre en compte les différentes contraintes sur les ressources disponibles, les aspects temps réel, l'ergonomie du système, la consommation d'énergie etc. Le concepteur vérifiera notamment que l'implantation des mécanismes liés à la concurrence des traitements de perception et d'action (défi 7) est conforme aux attentes, la conformité des processus d'auto-organisation (défi 10) etc.

### 5.2.3 Aperçu de l'architecture

Le simulateur MASH met en jeu de nombreux agents [1] qu'ils soient logiciels ou embarqués. Avec cet outil, les agents déployés sur leurs propres plateformes interagissent ensemble mais aussi avec les agents logiciels et l'environnement virtuel géré par le *gestionnaire d'environnement*. Faire collaborer des agents réels et virtuels n'est pas une fin en soit pour l'outil : son objectif est de permettre le développement fiable d'une société d'agents embarqués pour répondre à un problème donné.

Ajouter un nouveau type d'agent logiciel dans l'outil MASH revient à dériver une classe `Agent` pour implanter le comportement que l'on souhaite donner à l'agent et à choisir son modèle de batterie. L'incorporation d'un agent embarqué peut ne nécessiter que la redirection des émissions et réceptions de trames vers MASH.

Si le concepteur veut profiter de toutes les possibilités proposées par MASH pour la mise au point du SMA, il est alors nécessaire d'implanter dans les agents embarqués le protocole système MASH qui permet à l'agent de communiquer avec le simulateur via un agent proxy. Dans ce cas, il n'est plus nécessaire de rediriger l'émission/réception des trames de l'agent embarqué vers MASH, il suffit de mettre en place leurs notifications. Ainsi, si l'agent embarqué envoie un message de son protocole de négociation via son interface wifi, les autres agents embarqués pourront directement la percevoir tandis que la notification permettra à MASH de faire suivre la trame aux agents notamment virtuels qu'il considérera à sa portée.

La figure 5.4 présente le schéma de l'architecture simplifiée du simulateur et illustre l'utilisation des différents modèles utilisés.

**Le *Gestionnaire d'Agent Individuel*** Chaque agent possède ses propres modèles et sa propre architecture. Un agent peut être implanté sous la forme d'un agent logiciel (une classe java) ou d'un

1. Dans nos problèmes applicatifs, nos simulations comportent au maximum un millier d'agents.

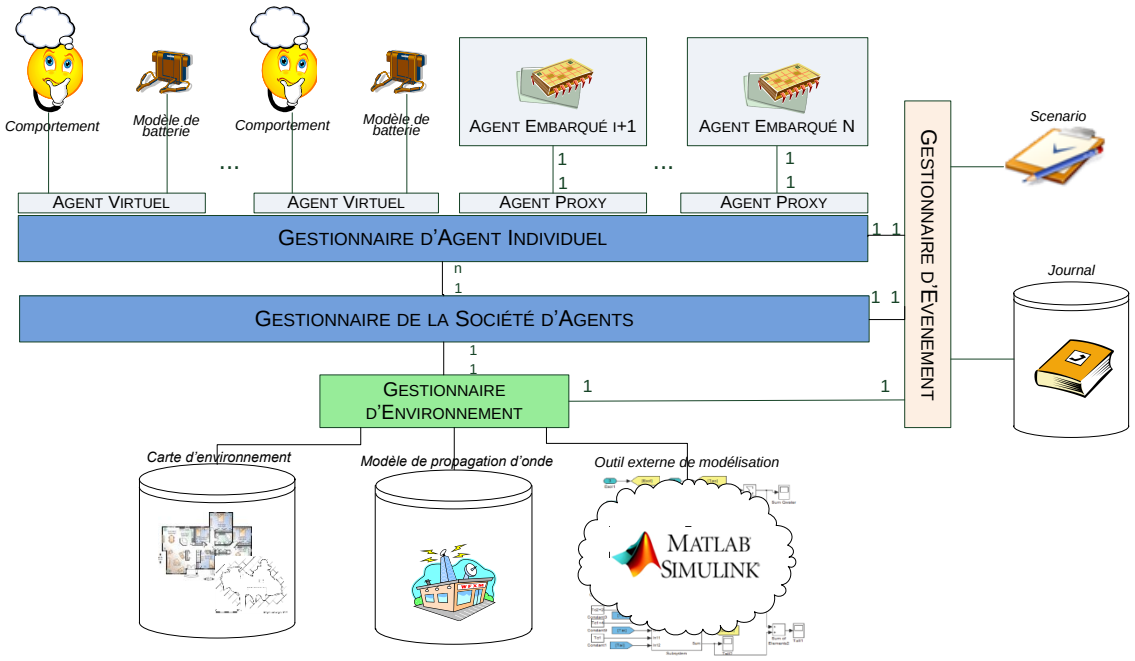



FIGURE 5.4 – Architecture simplifiée de MASH

agent embarqué. Le rôle du *Gestionnaire d'Agent Individuel* est d'uniformiser l'interaction des agents avec le composant *Gestionnaire de la Société d'Agents*, en d'autres termes de faire abstraction de la véritable nature de l'agent. Si un message est envoyé par un autre agent à l'agent embarqué, ce message sera redirigé vers son *agent proxy* qui exploitera l'interface réseau ou le port de la machine qui l'héberge (port série, USB etc.) afin d'atteindre l'agent embarqué : à tout message applicatif MASH impose qu'il soit associé une spécification de niveau bit. Si un scénario demande qu'un agent embarqué déclenche une action, son *agent proxy* lui transmettra la demande la déclenchement via un message dit système (spécification de niveau bit propre à MASH non dépendante de l'application conçue).

**Le composant *Comportement*** Ce module est dépendant de l'application : il spécifie le cycle de décision de l'agent. Dans le cas des agents réactifs, il s'agit généralement d'une implantation d'un simple graphe d'état-transition.

**Le *Gestionnaire de la Société d'Agents*** Ce module s'occupe de la mise en relation des agents entre eux et avec leur environnement. C'est ce module qui définit la localité d'un agent c'est-à-dire qu'il permet à un agent :

1. d'identifier les agents qui sont dans son champ de perception. Par exemple, quand un agent envoie un message, c'est ce composant qui recherche ceux qui sont à portée et pourront donc recevoir le message émis.
2. d'accéder aux valeurs de l'environnement que les capteurs sont capables de mesurer (et uniquement à celles-ci) ;
3. d'agir sur l'environnement avec ses effecteurs, en d'autres termes, de modifier les paramètres

du modèle de l'environnement.

**Le *Gestionnaire d'Environnement*** Il simule l'environnement des agents. A minima, ce modèle est composé :

— d'une *carte de l'environnement* qui le décrit sous la forme d'une grille 2D. Les positions des agents sont mémorisées au niveau de ce composant et non sur l'agent parce que dans de nombreuses applications, la plateforme sur laquelle est déployé l'agent ne propose pas de service de localisation. Si l'agent est équipé d'un périphérique de type *Global Positioning System (GPS)* le gestionnaire lui autorisera l'accès à cette information.

— d'un *modèle de propagation d'ondes* qui conditionne la perception des messages envoyés via des dispositifs de transmission sans fil. Ce modèle peut être simpliste (définition d'une portée fixe par dispositif) ou décrire plus précisément la diffusion de l'onde au travers de la grille 2D évoquée précédemment.

Il peut aussi incorporer :

— des modèles décrivant l'évolution de critères physiques directement liés à l'application comme l'évolution de la température d'une pièce dans un contexte "domotique". Sa mise en œuvre peut éventuellement nécessiter des outils externes de modélisation tels que Matlab/Simulink, Scilab, Labview...

**Le *Gestionnaire d'Événements*** La traçabilité de la simulation est assurée par ce composant. Les évènements systèmes (création/destruction d'un agent, envoi de trames...) sont générés par les gestionnaires décrits précédemment tandis que les évènements applicatifs (détection d'un incendie par un capteur, début d'une poursuite engagée par un agent prédateur...) sont générés par le code applicatif des agents en dérivant une classe `UserEvent`).

**Scénario** En ce qui concerne le traitement de scénario, le simulateur utilise l'API de réflexion de Java qui permet d'inspecter en cours d'exécution une classe, c'est-à-dire d'observer les attributs et les méthodes définies. Ainsi quand à une date donnée, un scénario demande l'exécution d'une méthode sur un agent, il n'est pas nécessaire que cette méthode ait fait l'objet d'une déclaration par l'utilisateur : il suffit que la méthode existe dans le code de l'agent.

### 5.2.4 Exemples d'utilisations

**Application mettant en œuvre le modèle MWAC** La figure 5.5 est une capture de différentes fenêtres utilisées lors de l'exécution d'un scénario.

Ces fenêtres permettent d'observer le système simulé à différents niveaux d'abstraction :

1. Niveau "société" :
   — la fenêtre à gauche de la figure 5.5 permet l'observation du système dans sa globalité (cette vue est personnalisable par le concepteur d'une solution). Elle montre une grotte dans laquelle a été déployé un SMA d'instrumentation sans fil implantant le modèle MWAC. Les disques colorés représentent des agents (rouge : agent représentant, vert : agent liaison, jaune : agent simple membre). Leurs accointances sont graphiquement matérialisées par les segments bleus.

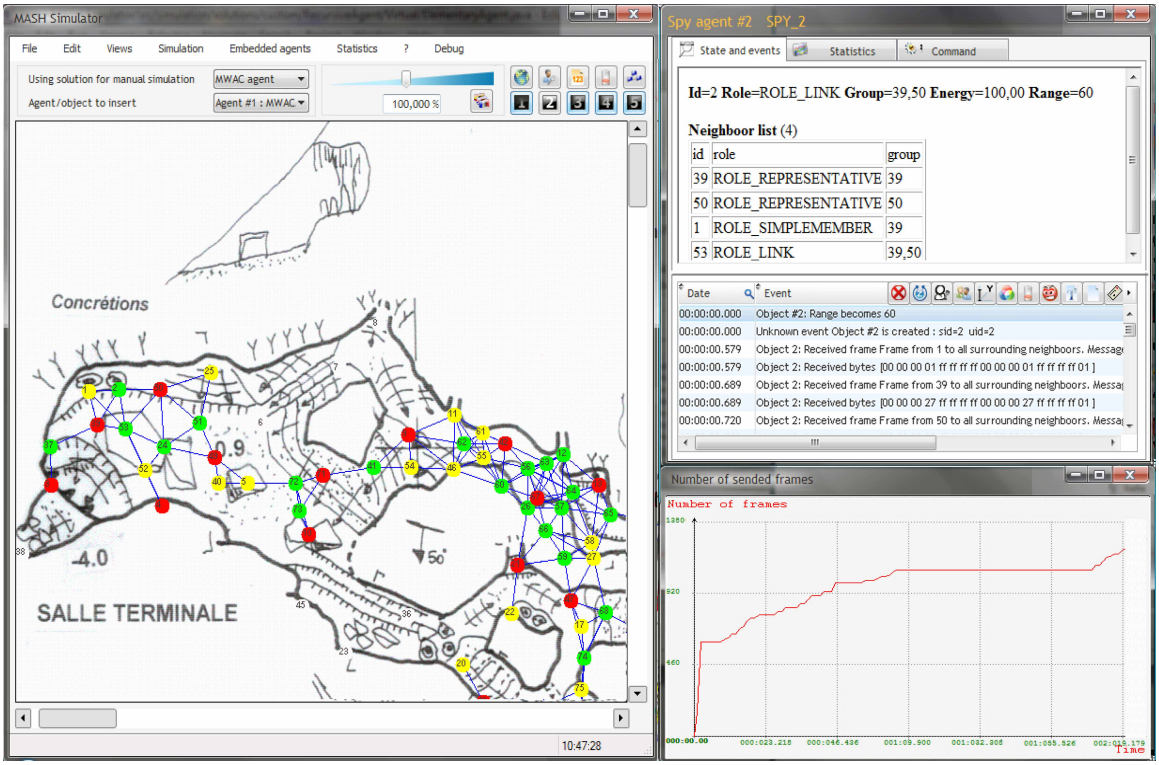



FIGURE 5.5 – Exécution d'un scénario d'instrumentation dans fil avec MASH

— la fenêtre en bas à droite de la figure 5.5 permet d'observer le systèmes selon des critères systèmes (`SystemCriteria`) ou définis par l'utilisateur en dérivant la classe `UserCriteria`. Le graphique fait ici apparaître le nombre de trames envoyées en fonction du temps.

2. Niveau "individu" :
   — la fenêtre en haut à droite de la figure 5.5 permet d'inspecter l'état interne d'un agent. Ainsi on voit que l'agent #2 (un agent embarqué) a le rôle liaison, qu'il est à son niveau d'énergie maximale et qu'il est en relation avec les agents #1, #39, #59, #53. Plus bas, on observe différents événements le concernant. La plupart des évènements systèmes ont été signalés par l'*agent proxy* tandis que les évènements applicatifs le sont uniquement par l'agent embarqué en utilisant le protocole système MASH. Par exemple, la quatrième ligne du tableau indique que l'agent #2 a reçu la trame suivante : `00 00 00 01 FF FF FF FF 00 00 00 01 FF FF FF FF 01`. Dans le protocole applicatif implanté par le concepteur de cette solution, il s'agit d'un message HELLO diffusé par l'agent #1 à tous ses voisins.

**Application colonie d'objets semi-vivants** La photographie en figure 5.6 est une mise en œuvre d'un jeu proie/prédateur. Cette application a bénéficié du soutien du CEA (financement de Emmanuel Jez et Rémy Thomas - DRT en Informatique).

Le vidéoprojecteur, relié à un ordinateur exécutant MASH, projette au sol l'environnement virtuel maintenu par le simulateur. L'agent embarqué (un robot construit à partir d'un FPGA Virtex 4 - Xilinx XC4VFX20) est un des prédateurs du jeu et poursuit une proie virtuelle que l'on voit

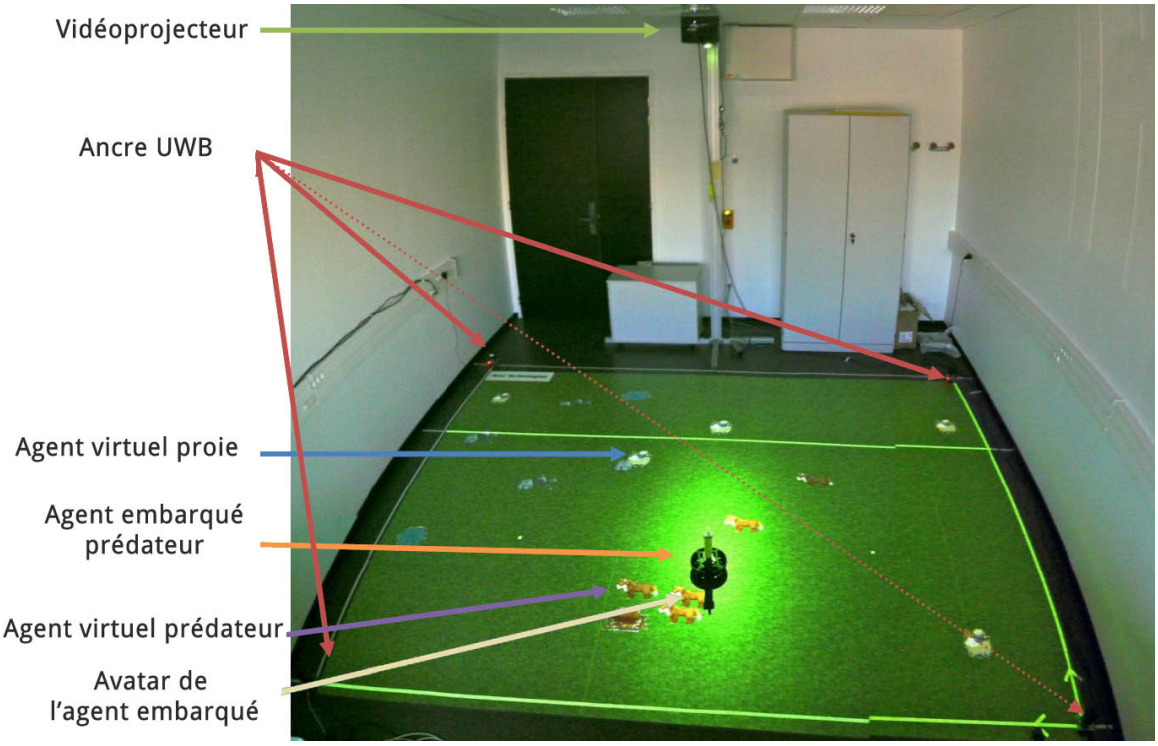



FIGURE 5.6 – Une implantation cyber-physique du jeu proie/prédateur avec MASH

projetée sous la forme d'un mouton. La position du robot est mesurée par le prototype de système de radiolocalisation que nous avons participé à concevoir en utilisant la méthode DIAMOND {Occello et al., 2008} (résultat du travail de recherche de Rémy Guillermin, étudiant de DRT que nous avons encadrés). Elle est injectée dans MASH et utilisée pour visualiser son avatar qui le représente dans la simulation. L'imprécision du système de mesure explique que cet avatar n'est pas complètement recouvert par le robot.

**Application à la supervision multi-niveau de réseaux d'instrumentation** MASH a été étendu pour permettre la vue à différents niveaux d'abstraction d'un même CCP. La figure 5.7 illustre l'utilisation de l'outil dans le contexte de l'observation d'une implantation d'un réseau d'instrumentation implantant ≡MASH. La figure 4.9 page 77 a été obtenue en agrégeant des vues obtenues avec MASH.

## 5.3 Discussion

Dans cette partie, nous nous concentrons sur les principales particularités de MASH. Pour chacune d'elles, nous présentons une revue rapide des travaux connexes et donnons un aperçu de sa mise en œuvre dans MASH.

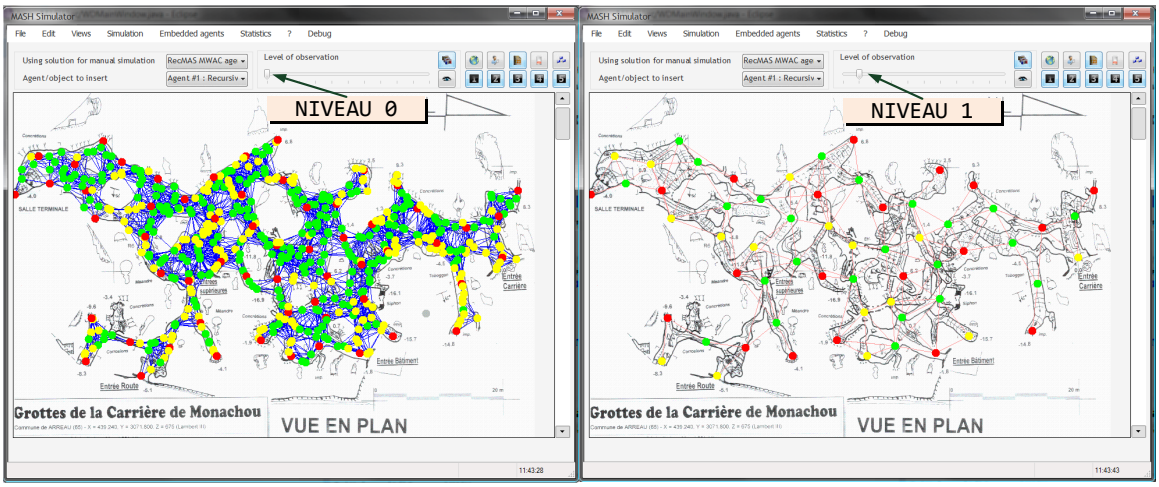



FIGURE 5.7 – Une utilisation de MASH pour l'observation et la supervision multi-niveau

### 5.3.1 Simulation de sociétés mixtes réelles/virtuelles

**Problème.** Lors d'une phase préliminaire de conception, il est plus confortable pour le concepteur de travailler sur un prototype purement virtuel (en simulation) du système. Lorsque le système virtuel semble répondre aux exigences, il faut alors construire les agents embarqués. Les adaptations d'implantation des comportements locaux des agents embarqués peuvent induire des déviations dans le comportement du système global. Permettre d'inclure dans les simulations des agents physiques participe à la maitrise de ces déviations.

**Travaux existants.** Des outils de simulation des réseaux sans fil et notamment dans le contexte des réseaux de capteurs, comme ceux abordés dans {Levis et al., 2003; Girod et al., 2004; Sobieh and Hou, 2003; Titzer et al., 2005}, permettent d'intégrer des nœuds matériels en se focalisant sur les couches de bas niveaux (simulation de modèles de batteries, simulation des couches physiques et liaison). Ils ne sont pas compatibles avec des propriétés comme l'ouverture et surtout avec le passage à l'échelle {Erciyes et al., 2011} rendant difficile l'évaluation d'une solution selon des critères liés à la complexité (défi 6). Nous avons proposé un état de l'art de ces simulateurs dans {Jamont and Occello, 2008}.

Dans des domaines de recherche plus proches, on trouve des simulateurs incluant des parties physiques mais ils ne visent pas non plus l'aide à la conception de systèmes embarqués décentralisés. Par exemple, dans {Weyns et al., 2005}, les auteurs utilisent des robots pour recréer des environnements virtuels à partir des observations physiques. L'objectif est de rendre des processus de planification plus pertinents. Les travaux multi-agents utilisant des tables tangibles {Kubicki et al., 2013; Thévin et al., 2014} peuvent être aussi considérés comme des systèmes cyber-physiques dans lesquels des agents en quelque sorte physiques et des agents logiciels interagissent. Certains simulateurs de vols {Guido and Montrucchio, 2006} utilisent des dispositifs physiques dans l'objectif de proposer une simulation globale du système la plus réaliste possible. Les équipements physiques ne sont en fait que des interfaces pour le simulateur.

**Nos propositions.** Dans notre outil, tous les agents embarqués dans le monde réel sont associés soit à des *agents proxy* qui les représentent dans la société simulée agissant comme des passerelles, soit des *avatars* qui étendent leurs comportements et leurs connaissances. Utilisés par le *Gestionnaire d'Agent Individuel*, une des fonctionnalités importantes qu'ils assument pour MASH, consiste à traduire les messages du monde virtuel en des messages physiques et vice versa. Ils peuvent aussi être utilisés pour comparer le comportement des agents embarqués avec le comportement qu'auraient eu les agents virtuels originels s'il les avaient remplacés.
Par exemple, la figure 5.8 montre le rôle attendu (celui d'un agent virtuel qui simule le comportement que l'on souhaite donner à l'agent embarqué) et le rôle obtenu par l'agent embarqué. A $t_r$ `= 152.45s`, nous pouvons voir que l'agent embarqué nécessite plus de temps pour choisir son prochain rôle ce qui le conduit à adopter un rôle inattendu (simple membre) pendant un court laps de temps à $t_r$ `+ dt`. Ce délai est dû au temps plus important que nécessite le traitement des messages reçus par l'agent.

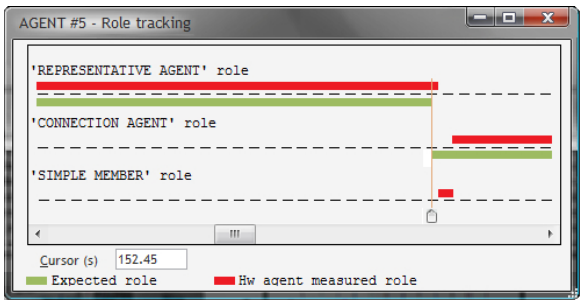



FIGURE 5.8 – Suivi du rôle d'un agent

### 5.3.2 Simulation utilisant des modèles physiquement réalistes

La plupart des travaux de l'intelligence artificielle distribuée et des SMA qui semblent s'intéresser à construire des systèmes dans des environnements physiques ne sont en fait souvent motivés que par des considérations purement logicielles. Un bon exemple est le champ applicatif "domotique" qui a vu se multiplier des contributions : conception de protocoles d'interaction pour négocier des préférences utilisateurs {Hsu and Wang, 2008}, prise de décision collective et résolution distribuée de problème en prenant en compte des problèmes relatifs à la consommation d'énergie {Begg and Hassan, 2006; Abras et al., 2010; Nishiyama and Sawaragi, 2011}, extraction de données afin d'établir des liens entre des situations passées et présentes {Galton, 2006} etc. Les outils utilisés dans ce contexte pour simuler ces solutions sont développés par les auteurs et ne sont pas prévus pour permettre la mise en œuvre de modèles réalistes de l'environnement (comme ceux décrits dans {Carolis et al., 2005; O'Neill et al., 2005; Kim et al., 2006; Nguyen et al., 2010}).

Dans cette section, nous aborderons trois types de modèles cruciaux pour la conception des systèmes cyber-physiques :

— Les modèles d'environnements applicatifs qui décrivent la physique des processus mise en jeu

dans l'application qu'un concepteur veut construire. Il peut s'agir du modèle thermique d'une maison, du modèle d'écoulement d'un liquide en surface libre, d'un modèle de freinage qui vise à rendre plus réaliste l'évaluation d'évitement de collisions de robots. Ces modèles sont souvent dépendants de l'application.

— Les modèles de consommation d'énergie qui permettent d'estimer le coût en énergie de la sollicitation d'un effecteur, d'un actionneur, d'un module de transmission radio-fréquence etc. Ils permettent ainsi de simuler l'évolution de l'énergie disponible sur un agent.

— Les modèles de propagation d'ondes qui dans certaines applications comme celle de la radio-localisation en intérieur peuvent revêtir une grande importance.

#### 5.3.2.1 Modèles d'environnements applicatifs

**Problème.** Lorsqu'on s'intéresse à des problèmes de supervision ou de commande mettant en œuvre des processus physiques, il est important de disposer de modèles réalistes qu'ils soient analytiques (modèle de connaissance construit à partir des lois de la physique par exemple) ou expérimentaux (modèle de représentation construit par identification) tant leur impact sur les résultats d'une simulation est important. Ces modèles complexes sont mis en œuvre avec des outils spécialisés dans comme Labview {Fairweather and Brumfield, 2011}, Matlab{Moore, 20011} et Scilab {Braatz, 2014}.

**Travaux existants.** Avec d'un côté l'intérêt grandissant de l'automatique pour des approches "multi-agents" {Choi et al., 2009} et de l'autre, l'augmentation de l'importance des applications relatives aux systèmes embarqués pour les SMA, l'utilisation des outils d'une communauté par l'autre émergent. Par exemple, LabVIEW a été utilisé pour implanter des SMA {Ponci et al., 2005; Conte et al., 2009}, les agents étant conçus comme des instruments virtuels. Les auteurs perdent ainsi de nombreux avantages des simulateurs multi-agents : simulation à large échelle, facilité d'utilisation de modèles spécifiques aux multi-agents... Plus récemment, dans le contexte de la commande distribuée, des SMA sont implantés sous Matlab/Simulink comme dans {Mendes et al., 2010}. Dans le domaine de SMA, une extension de Jade permettant l'interaction avec Simulink a été proposée {Robinson et al., 2010}.

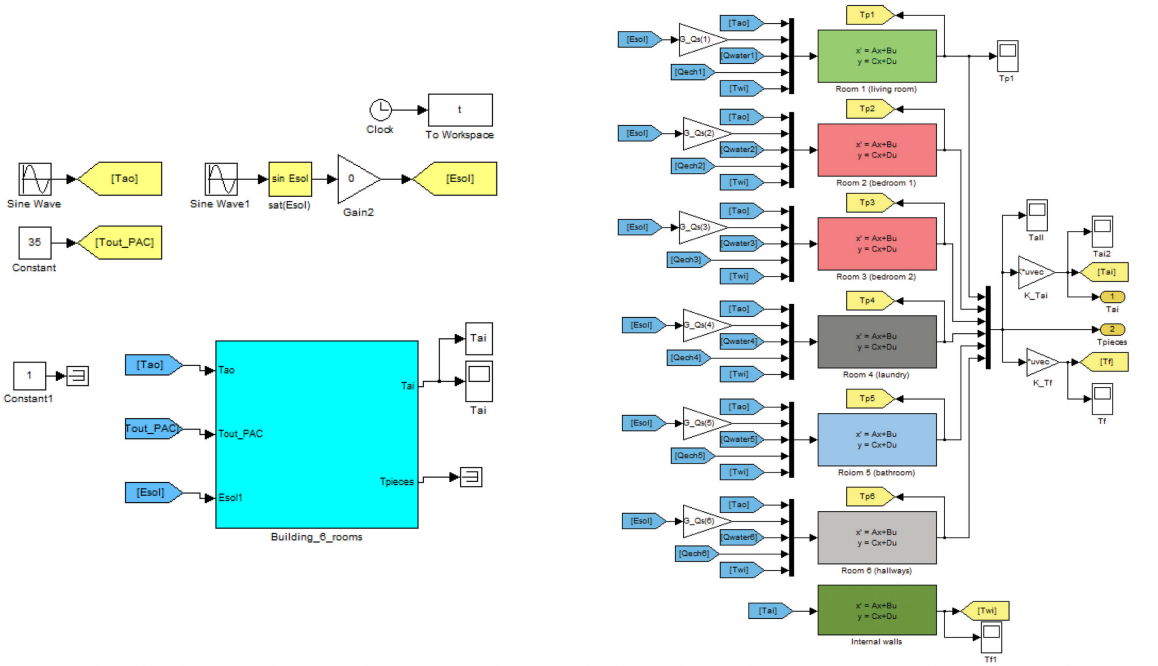
**Nos propositions.** La création d'un tel modèle nécessite d'avoir une connaissance sérieuse de la physique de l'objet d'étude. Matlab est sans aucun doute l'outil le plus approprié pour mettre en œuvre le modèle d'un système physique complexe. Simulink est une extension de Matlab permettant une programmation visuelle. Un modèle Simulink se présente sous la forme d'un schéma bloc (un bloc étant une fonction mathématique ou une composition décrivant un système dynamique) permettant de résoudre des équations algébriques ou des équations différentielles ordinaires. Ce découpage en blocs permet une meilleure intelligibilité du modèle global et une réutilisation de ses parties.

Dans le cadre d'une action transversale de recherche, nous avons collaboré avec l'équipe d'automatique de notre laboratoire notamment en co-encadrant des étudiants de master. Nous nous sommes intéressés à la gestion du confort thermique des bâtiments {Jamont et al., 2011}. La modélisation de l'évolution de la température d'un bâtiment vu comme un ensemble de pièces était

nécessaire pour confronter nos approches de commande. Le modèle que nous avons créé (masters de Farid Jebahi et Antoine Saillot) permet d'instancier des modèles thermiques de maisons en deux étapes :

1. On modélise une pièce indépendamment des autres en décrivant ses composants (murs externes, fenêtres, portes donnant sur l'extérieur, chauffages etc.) qui ont eux aussi leurs propres modèles ;
2. On établit les liens entre les pièces en précisant les éléments partagés (murs internes, plafond si plusieurs étages, portes etc.)

Le résultat est un modèle modulaire mis en œuvre dans Matlab/Simulink. Certaines parties de ce modèle sont illustrées en figure 5.9. On peut observer le modèle thermique d'une maison de plain-pied composée de 6 pièces (figure 5.9a). La représentation visuelle du calcul des températures des chambres et des parois internes qui composent le bâtiment est visible en figure 5.9b. Le détail des équations sous forme matricielle (représentation d'état) est donné dans {Jamont et al., 2011}.




a. Modèle d'une maison à 6 pièces b. Modèle de l'évolution de la température (pièces et murs internes)

FIGURE 5.9 – Partie du schéma bloc modélisant l'environnement physique "Maison de 6 pièces"

Pour utiliser un tel modèle, MASH peut interagir avec Matlab/Simulink de deux manières :

1. en utilisant un mode de communication client-serveur :
   Cette solution nécessite l'extension *Simulink Coder* (anciennement *Real-Time Workshop*). Il permet soit de dériver un modèle Simulink en un code source C/C++, soit d'exécuter un code écrit dans un de ces langages pendant la mise en œuvre d'un diagramme Simulink. Nous utilisons cette deuxième fonctionnalité pour permettre l'interaction avec MASH : Simulink agit en tant que serveur de communication et MASH se connecte en tant que client pour accéder aux paramètres du modèle physique.

2. par échange de fichiers *Extensible Markup Language (XML)* :

   Parce que la solution précédente nécessite d'acheter un module spécifique, nous avons développé un script Matlab (travail de Paul Gery durant son master) qui permet une communication avec MASH via deux fichiers XML. Ces fichiers sont par défaut :

   — `data.in.xml` qui contient les paramètres en entrée du modèle Simulink et donc ce sur quoi les agents de MASH agissent,

   — `data.out.xml` qui contient les paramètres en sortie du modèle à savoir ce que les agents dans MASH peuvent percevoir.

   Un exemple de contenu de ces fichiers partagés est donné en figure 5.10. MASH utilise donc un algorithme très simple (Algorithme 1) afin de mettre à jour les données relatives à l'environnement dans lequel évoluent les agents.

**Algorithm 1** Intégration des paramètres du modèle physique Matlab/Simulink dans MASH

**repeat**
  Read environment values in `data.out.xml`
  Update MASH virtual environment
  Agents act
  Write controllable values by agents in `data.in.xml`
**until** simulation.isRunning()

```
<?xml version= "1.0">
<!-- Written on 24-Aug-2011 10:52:41 using the XML Toolbox for Matlab -->
<root>
   <data1>
     <name>simulated_time</name>
     <type>float</type>
     <value>30</value>
     <unit>day</unit>
     <description>Simulation time : day</description>
      <direction>out</direction>
   </data1>
    ...
   <data17>
     <name>Tai_03_03</name>
     <type>float</type>
     <value>32.678271140525</value>
     <unit>celcius</unit>
     <description>Ambient air temperature </description>
     <location>room 3</location>
     <direction>out</direction>
   </data17>
    ...
</root>
```

FIGURE 5.10 – Exemple de fichier XML échangé entre Matlab/Simulink et MASH

#### 5.3.2.2 Modèles liés à l'énergie disponible

**Problème.** Dans la plupart des simulateurs de systèmes intelligents décentralisés, il n'y a pas à proprement parlé de modèles liés à la consommation et au stockage de l'énergie. Pourtant, modéliser l'énergie disponible d'un agent permet de juger de la bonne prise en compte (dans leurs cycles de décision des agents), des contraintes non fonctionnelles relatives à la gestion de l'énergie disponible (défi 4]). Ces modèles peuvent être utilisés par un simulateur pour estimer l'autonomie en énergie d'un agent ou par un agent lui-même pour mesurer l'impact d'un plan qu'il construit sur ses réserves d'énergie. Cependant, il est communément admis que les modèles d'état de charge des batteries ont une marge d'erreur allant de $\pm 10\%$ à $\pm 20\%$.

**Travaux existants.** Il est difficile de définir un modèle générique de consommation d'énergie tant les comportements de charge et de décharge de ces batteries et les profils de consommation des capteurs et actionneurs sont différents. Aussi, il est intéressant qu'un outil fournisse plusieurs modèles paramétrables plus ou moins complexes suivant l'importance que revêt cette estimation de l'énergie. La façon la plus simple pour modéliser la dépense énergétique consiste à définir pour chaque agent la capacité initiale de la batterie et de considérer cette dernière comme une source linéaire de courant. Il existe cependant de nombreux autres modèles {Park et al., 2001; Handy and Timmermann, 2003}.

**Nos propositions.** Dans MASH, nous avons défini une structure relativement générique qui permet de mettre en œuvre différents modèles de batterie. Nous utilisons souvent le modèle de décharge linéaire. La capacité $C$ d'une batterie après un laps de temps $td$ peut être exprimée par l'équation suivante : $C = C' - \int_{t=t_0}^{t_0+t_d} I(t)dt$ avec $C'$ est la capacité de la batterie à l'instant précédent et $I(t)$ est le courant instantané consommé à l'instant $t$ par l'ensemble des matériels de l'agent. Le concepteur soit donc associer des consommation instantanées à certaines opérations. Pour cela nous définissons des états en précisant la consommation de courant qu'elle entraîne :

- `émission radio` (19.5 mA),
- `réception radio` (21.8 mA),
- `cpu actif` (1.8 mA),
- `cpu en mode sommeil` (1.2 mA).

Un concepteur peut personnaliser les états et les valeurs afin de les faire correspondre à son matériel et couvrir toute la gamme de capteurs et d'actionneurs utilisés.
MASH pouvant interagir avec des outils de simulations externes, un modèle de batterie implanté par exemple sous Matlab peut aussi être utilisé.

#### 5.3.2.3 Modèles de propagation d'ondes

**Problème.** Si pour la plupart des applications, l'utilisation d'un modèle particulier de propagation de l'onde n'est pas indispensable, il peut cependant s'avérer critique dans certaines applications. Nous présentons ici l'utilisation de ces modèles dans MASH.

**Travaux existants.** Les outils dédiés à la simulation des réseaux sans fil comme NS2 {Issariyakul and Hossain, 2014} proposent de nombreux modèles qui permettent de caractériser l'impact d'environnement spécifique (villes, intérieur d'une maison...) sur la propagation des ondes. Ces modèles permettent aussi d'agir sur la modulation utilisée jusqu'aux débits, taux d'erreur binaire etc.
Dans le contexte de l'intelligence artificielle distribuée, des modèles aussi précis n'existent pas. Souvent, l'une des deux hypothèses suivantes est faite :

— la couverture de la zone géographique dans laquelle évoluent les agents par les points d'accès sans fil est supposée totale,

— la propagation des ondes est considérée uniforme et, en conséquence, le concepteur définit une portée d'émission pour chacun des agents (généralement une seule et même valeur pour l'ensemble des agents).

Les modèles de consommation d'énergie associés aux traitements de ces ondes (génération ou interprétation) sont souvent mal compris par les spécialistes des logiciels : l'énergie consommée pour retrouver l'information véhiculée par une onde est jugée négligeable par rapport à celle utilisée pour la générer alors que dans de nombreux cas, elle est en fait supérieure {Fourty et al., 2012} (illustration donnée en figure 5.11).

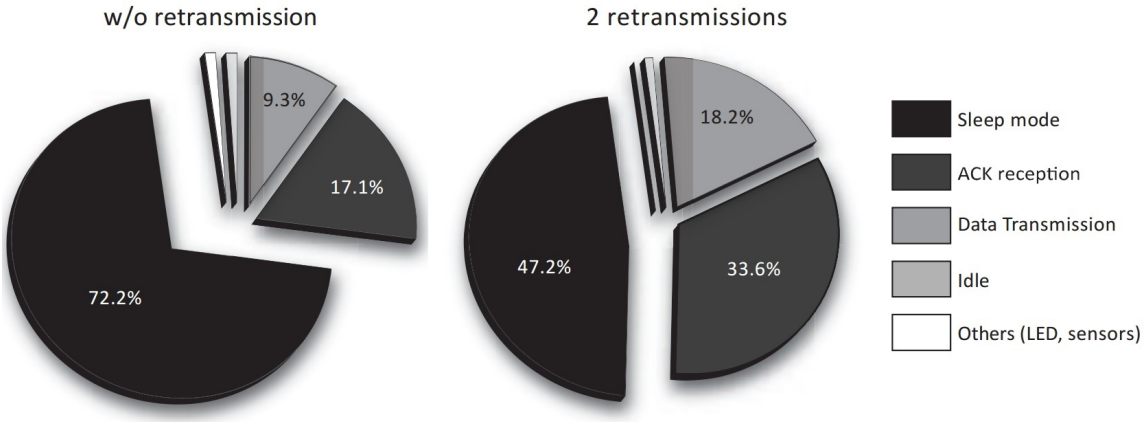



FIGURE 5.11 – Consommation de nœuds relevée expérimentalement sur 2 scénarios d'instrumentation, figure tirée de {Fourty et al., 2012}

**Nos propositions.** Parce qu'il est possible d'avoir plusieurs simulations Matlab concurrentes, la description de la propagation de l'onde peut aussi être modélisée dans l'environnement Matlab.
S'il n'est pas nécessaire d'utiliser un modèle spécifique, MASH propose une implantation basée sur l'équation de Friis. Cette équation décrit l'atténuation d'une onde en espace libre : $P_r = G_r.G_e.\left(\frac{\lambda}{4\pi d}\right)^2.P_e$ avec $P_r$ la puissance du signal reçue par un agent émis par un autre agent générant une onde de puissance $P_e$, $G_r$ et $G_e$ les gains linéaires des antennes utilisées à l'émission et à la réception, $d$ la distance séparant les agents, $\lambda$ la longueur de l'onde. A partir de l'estimation de $P_r$, MASH estime la probabilité d'une bonne réception du message. Ce modèle utilise donc les positions mémorisées dans le *Gestionnaire d'Environnement*.

Nicolas Fourty, spécialiste dans la gestion d'énergie dans les réseaux de capteurs qui a rejoint

notre laboratoire en 2008, va implanter les modèles qu'il obtient expérimentalement dans l'outil.

## 5.4 Conclusion

Ce chapitre a présenté les travaux autour d'un outil permettant d'accompagner le cycle de vie du développement d'un CCP. Même s'il a été conçu pour exploiter au mieux le cycle de vie en spirale de DIAMOND, il peut être vu comme un outil générique utilisable avec d'autres approches multi-agents.

### 5.4.1 Un outil pour construire des CCP

Construire des CCP nécessite de concevoir et d'ajuster le comportement d'agents embarqués. MASH permet la cohabitation d'agents simulés et d'agents embarqués dans une même étape de la simulation du CCP que l'on souhaite concevoir.

L'un des principaux avantages de MASH est de permettre une relative maîtrise de l'impact de la modification d'un comportement local d'agent embarqué sur le comportement collectif du système global. La mise en œuvre de modèles réalistes des dimensions physiques du CCP permet une meilleure exploration de l'espace des solutions d'un problème car elle permet de prendre en compte au plus tôt des contraintes non fonctionnelles comme celles liées à l'autonomie d'énergie.

Cet outil a bénéficié de retours d'expériences par des informaticiens et des automaticiens de notre laboratoire. Des étudiants de master d'automatique que j'ai encadrés avec Eduardo Mendès l'ont notamment utilisé dans le contexte de la domotique. MASH est peu diffusé car un important effort reste à faire pour créer sa documentation. Aussi, seuls nos partenaires de longue date (par exemple les équipes de Mouloud Koudil à Alger et de Félix Ramos à Guadalajara) les utilisent aujourd'hui.

### 5.4.2 De nouveaux défis à relever

Les applications relevant des CCP sont des systèmes complexes qui nécessitent souvent plusieurs expertises pour leur mise en œuvre. Pour chaque expertise, les spécialistes disposent souvent d'outils dédiés. Il n'est pas envisageable qu'un seul simulateur soit capable de toutes les intégrer.

Aussi si on ne souhaite pas limiter l'utilisation de MASH aux seuls membres de notre laboratoire ou à nos grands partenaires, il devra être capable de travailler avec d'autres simulateurs sans que les utilisateurs soient obligés de modifier son architecture. Des modèles de synchronisation entre ces outils seront alors nécessairement partagés.

# Chapitre 6

# Conclusion et Perspectives de recherche

Ce chapitre présente la synthèse des contributions exposées tout au long du manuscrit, les travaux que je mène aujourd'hui et la direction dans laquelle j'oriente mon projet de recherche.

## 6.1 Bilan

Mes travaux se sont particulièrement intéressés à l'articulation des objets physiques (niveau local) dans un collectif mixte logiciel et matériel (niveau global). Le problème de cette articulation, rencontré habituellement dans les systèmes multi-agents, est rendu plus difficile en raison de sa dimension physique. Tout d'abord, les agents physiques ont des ressources limitées. La puissance de calcul et les capacités de stockage étant moins importantes que dans le cadre des systèmes informatiques classiques, utiliser des modèles existants provenant de l'intelligence artificielle nécessite souvent de les aménager. Ensuite, la dimension physique du système global, et notamment l'environnement partagé avec l'humain, augmente le nombre et la criticité des contraintes non fonctionnelles qu'il convient de prendre en considération.

Concevoir des systèmes embarqués distribués, coopérants, ouverts et à intelligence décentralisée est un problème difficile qui ne peut être résumé à celui du déploiement des agents d'un SMA sur des dispositifs embarqués : il est nécessaire d'utiliser des modèles et des méthodes multi-agents spécifiques pour prendre en compte toutes les contraintes du monde physique.

Nous avons développé une méthode complète d'analyse et de conception pour construire des CCP à l'aide de SMA.

**Une démarche méthodologique expérimentale** Notre démarche méthodologique consiste en un ensemble ordonné de phases et d'étapes afin de guider le concepteur d'une solution embarquée distribuée et à intelligence décentralisée. Cette démarche part du recueil des besoins fonctionnels et des contraintes extra-fonctionnelles jusqu'au déploiement du système conçu.
Au niveau du cycle de vie, les principales originalités de nos contributions sont l'utilisation d'un cycle

en spirale, qui renforce les liens entre les processus et permet des itérations au niveau des phases, et la prise en compte de la dimension physique des agents en unifiant la construction des composantes logicielles et matérielles.

En terme d'analyse, nous proposons une étude des modes de marche et d'arrêt qui permet une structuration du fonctionnement global du CCP en identifiant ses différents contextes. Nous pouvons noter aussi l'utilisation de diagrammes de contexte afin de caractériser l'agent par rapport à son environnement physique et permettre au concepteur d'identifier les données et les événements du monde réel qui lui permettront de construire la représentation du monde qu'auront les agents.

Concernant la conception et l'implantation, nous avons proposé l'utilisation de composants comme unité de production pour permettre la simplification du partitionnement logiciel/matériel. Des critères pour le partitionnement des entités du système ont été proposés.

**Des modèles multi-agents opérationnels** Afin de relever différents défis posés par la réalisation de CCP, nous avons proposé deux architectures d'agent. Elles permettent l'intégration dans les agents des différentes dimensions de la décomposition multi-agent du problème. *eASTRO* est une architecture qui respecte les limitations de ressources des systèmes embarqués lors de la mise en œuvre de solutions décentralisées. *Avatar* est une architecture qui permet d'outrepasser les limitations de ressources des objets physiques et des systèmes embarqués en profitant des données et services hébergés par le Web.

Nous avons aussi proposé des modèles de collectif qui permettent de simplifier le travail du concepteur lorsqu'il souhaite doter le système global de capacités de communication, de coopération ou de maintien d'intégrité fonctionnelle. Par exemple, MWAC, TrustedMWAC et AntMWAC permettent de soulager le concepteur d'une partie des problèmes que soulève la mise en œuvre de la coopération dans les CCP. ≡MWAC répond au problème de l'intelligibilité des informations portées par les agents.

Des modèles ont été aussi été proposés pour produire et maintenir des services collectifs en environnements changeants.

**Des contributions outillées et validées** Les différentes activités de notre méthode sont supportées par des outils graphiques et un outil logiciel qui permet de maîtriser la déviation entre le comportement global obtenu en simulation et celui attendu lors de son réel déploiement. La plateforme MASH permet la mise au point, le test, l'exécution et le débogage des CCP. Les originalités des contributions autour de ces outils proviennent de notre volonté de rester proche des déploiements réels et donc des contraintes physiques.

En plus de coopération informelle avec divers collègues, l'ensemble de nos contributions a bénéficié de nombreux retours d'expérience de par leur utilisation dans plusieurs projets (résumés dans le tableau 6.1). Ces projets ont de plus permis de valider nos propositions.

| Nom du projet | Financement | Partenaire privilégié | Thème | Publication liée |
|---|---|---|---|---|
| ACLIRSYS | ANR | MACSY | Domotique | {Jamont et al., 2011} |
| ASAWoO | ANR | LIRIS | Web des objets | {Mrissa et al., 2015} |
| ENVSYS | Industriel | CREATIME | Instrumentation | {Jamont et al., 2010} |
| KURASU | Industriel | CEA-LETI | Robotique | {Jez, 2011} |
| METALISM | ECOS | CINVESTAV | Ingénierie des SMA | {Razo et al., 2010} |
| PALETTE | Industriel | EREIMA | Robotique | {Jamont et al., 2014b} |
| PULSER | EU | CEA-LETI | Radiolocalisation | {Occello et al., 2008} |
| SIET | Industriel | TMM Software | Soins à domicile | {Mercier et al., 2013a} |
| TASSILI | PHC | LMCS-Alger | Instrumentation | {Hamani et al., 2011} |
| VAICTEUR AIR2 | Industriel | MACSY | Domotique | {Jamont et al., 2011} |

TABLE 6.1 – Projets ayant permis des retours d'expérience sur nos contributions.

## 6.2 Vers une ingénierie des collectifs cyber-physiques

Cette section présente les perspectives de mon travail qui s'inscrivent dans la continuation des travaux que j'ai précédemment exposés.

**Intégration des exigences non fonctionnelles dans les agents** Si nous avons présenté le recueil des exigences non fonctionnelles (étude des modes de marche et d'arrêt), nous n'avons pas abordé l'intégration de ces exigences dans les agents.
Pour intégrer ces contraintes dans la boucle de décision des agents, notre approche actuelle découle naturellement de l'étape de recueil : elle consiste à définir les différents comportements associés aux différents modes et de les commuter en fonction des perceptions locales.
Afin d'être plus tolérant aux situations non prévues, avec Clément Raïevsky [1], nous avons commencé à travailler sur une façon plus originale d'intégrer ces exigences au niveau collectif. Nous proposons de doter le système multi-agent d'un état de *vigilance* partagé et implicite. Ainsi, dans ce contexte, nous nous attachons à répondre à trois questions :

1. Comment générer un état émotionnel au niveau individuel ?
2. Comment l'état émotionnel généré va-t-il se propager ?
3. Comment l'information sur l'état émotionnel des autres membres du groupe va influencer le comportement des agents ?

Ces questions encore ouvertes font l'objet de travaux que nous poursuivons ensemble.

**Architectures de SMA et d'agents fortement contraints** Certaines applications, comme c'est le cas par exemple dans le domaine du transport et particulièrement dans l'avionique, contraignent très fortement le dimensionnement des SMA. Les besoins vont bien au-delà des architectures traditionnelles. Il faut alors s'intéresser à implanter des architectures d'agent et de SMA sur des *Systems on Chip (SoC)* ou des *Multi-Processor Systems on Chip (MPSoC)*.

Avec Virginie Fresse (Maître de conférences à l'Université de Saint-Étienne, LHC) nous nous intéressons à la conception de tels systèmes cyber-physiques critiques. Afin de tendre vers des SMA intégrant des contraintes temps-réel que nous n'avons pas réellement pris en compte jusqu'ici, il sera nécessaire de développer des architectures d'*Agent on Chip (AoC)* et de *MultiAgent on Chip*

1. Post Doc qui a rejoint notre équipe en tant que Maître de Conférences en septembre 2015

*(MASoC)*. En ce sens, nous avons déjà encadré des étudiants ingénieurs sur des problématiques liées au CloudFPGA qui visent la mutualisation de chaînes de développement et de cartes FPGA pour plusieurs utilisateurs.

**Un service transversal de gestion de la confiance** Avec Laurent Vercouter (Professeur à l'INSA de Rouen, LITIS) nous poursuivons notre collaboration sur les modèles de confiances adaptés aux CCP. Afin de proposer un service transversal de la gestion de confiance et ne pas la confiner aux aspects les plus logiciels, nous développons de nouveaux travaux conjointement avec Nicolas Fourty (Maître de Conférences, LCIS) et Adrien van den Bossche (Maître de Conférences, IRIT) spécialistes des couches basses des réseaux sans fil.

**Commande et supervision des collectifs cyber-physiques** Afin de pouvoir contribuer dans le domaine de la commande des collectifs cyber-physiques, j'ai initié des travaux avec l'équipe d'automatique du laboratoire LCIS. MASH était d'ailleurs un outil compatible avec nos différentes approches. Il nous a permis d'évaluer nos premières contributions de lois de commandes réellement [2] décentralisées {Pham et al., 2014; Raievsky et al., 2014}. Les projets ANR ACLIRSYS (2012/2016) et ADEME Vaicteur AIR2 (2012/2016) vont nous permettre de proposer des modèles génériques de commande décentralisée dans le contexte de l'habitat.

**Mutualisation d'expertises outillées** Afin de mutualiser différentes expertises via leurs outils de simulation, nous proposons avec Flavien Balbo (Professeur à l'École de Mines de Saint-Étienne) de concevoir un modèle de couplage entre modèles de simulation hétérogènes en fonction de différents niveaux d'abstraction (à l'échelle de différents écosystèmes). Notre contribution pratique consistera en un système d'aide à la décision permettant la compréhension de CCP plus complexes.

Le défi que nous adressons est celui de permettre l'observation d'un même système selon différentes perspectives (environnement, régulation, sécurité,...) avec pour chacune un niveau d'abstraction variable allant du microscopique au macroscopique. Il s'agit également de permettre l'utilisation simultanée des différents écosystèmes (les différents niveaux d'abstraction et les différentes perspectives). S'il peut exister par dimension et par niveau d'abstraction des modèles de simulation et des simulateurs dédiés, c'est leur utilisation conjointe qui permet de donner une compréhension du système sur un centre d'intérêt précis. Afin de permettre cette compréhension, nous nous appuierons sur deux domaines scientifiques :

1. le raisonnement spatial et temporel afin de formaliser les concepts nécessaires à la définition d'un référentiel spatio-temporel commun. Ce référentiel nous permettra à la fois de définir les règles de transition entre dimensions, entre niveaux d'abstraction et également entre les zones spatio-temporelles où les transitions doivent être réalisées ;
2. les systèmes multi-agents qui par leur capacité d'intégration, de coordination et d'adaptation permettront de synchroniser dynamiquement les simulateurs.

Khadim Ndiaye va commencer une thèse sur cette thématique (financement par la région Rhône-Alpes, ARC7) en octobre 2015.

2. Des systèmes purement distribués à intelligence centralisée sont quelquefois nommés à tort décentralisés dans la communauté de l'automatique.

## 6.3 Vers le couplage de collectifs cyber-physiques

Je propose dans cette section des développements originaux qui permettent de définir des objectifs à plus long terme.

### 6.3.1 Le couplage des collectifs

La coopération permet à des entités autonomes d'atteindre des objectifs communs en partageant leurs connaissances et/ou leur savoir-faire. Dans le cadre des CCP, un même environnement physique peut être partagé par plusieurs SMA. L'interaction et la coopération de ces SMA est alors rendue possible voire souhaitable dans la mesure où elle peut permettre à un des systèmes d'atteindre ses propres buts en exploitant de manière opportuniste les compétences disponibles dans d'autres SMA. Cette coopération apporte ainsi une valeur ajoutée aux différentes organisations et aux différents utilisateurs du CCP : qualité des services rendus, résilience, etc.

Pour illustrer ce type de coopération, considérons trois SMA qui œuvrent à la surveillance des forêts dans une perspective de détection et de lutte contre les incendies [3]. Comme l'illustre la figure 6.1, le premier SMA (SMA1 : surveillance au sol) est constitué d'agents capteurs qui doivent remonter des événements comme la variation brutale de la température vers une station de base. La station de collecte fiabilise ces informations et déclenche des alarmes au niveau de son organisme administrateur. Le deuxième SMA (SMA2 : surveillance aérienne) est constitué de drones équipés de caméras infrarouges. Ils survolent les espaces boisés à la recherche d'éventuels départs de feu et participent à leur extinction le cas échéant via un dispositif de dispersion de poudre extinctrice. Le troisième SMA (SMA3 : suivi d'équipes d'intervention au sol) est constitué d'agents associés aux pompiers instrumentés, aux véhicules qu'ils utilisent et à un . A partir des informations remontées par les pompiers, le Quartier Général (QG) contrôle le déploiement des équipes.

Chacun de ces SMA est opérationnel mais administré par une organisation différente : il accomplit ses tâches indépendamment des autres en ne comptant que sur ses propres compétences. L'objectif global de chacun des trois SMA est cependant lié à la lutte contre les incendies des forêts.

Considérons maintenant les services collectifs internes produits par chacun des SMA et indépendamment des autres :

— SMA1 : acheminement d'un message d'un capteur à la station de base.
— SMA2 : acheminement d'un message d'un drone à un autre ou d'un drone à son centre de commandement, extinction de feu.
— SMA3 : acheminement de messages entre agents (pompiers, véhicules, QG), extinction du feu.

Supposons qu'un agent capteur du SMA1 détecte un départ de feu. N'étant pas dans le voisinage de la station de collecte, il doit solliciter le service collectif d'acheminement de message interne à son SMA. La destruction d'un capteur a entraîné la partition du réseau de communication sous-jacent au SMA1 en deux sous-réseaux distincts (illustration en figure 6.2) rendant le service collectif défaillant. Un agent du SMA1 détecte ce problème interne de communication mais le SMA1 n'a pas les capacités d'adapter sa topologie pour pallier le problème. Une solution possible serait qu'une partie des agents de SMA2 maintienne une passerelle de communication entre ces deux partitions.

3. Le scénario et les figures sont tirés de {Khenifar Bessadi et al., 2015}

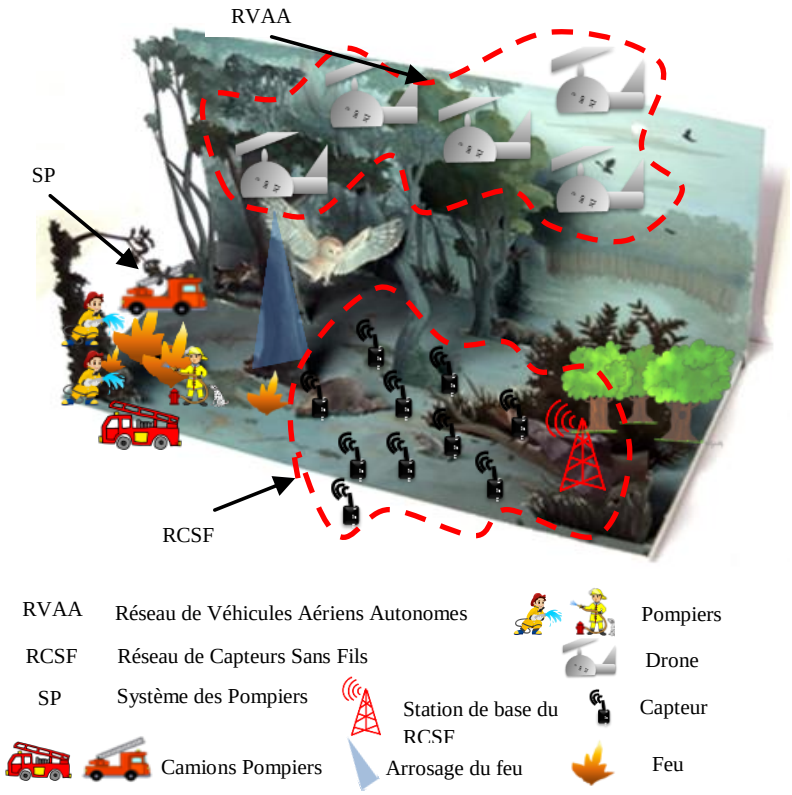



FIGURE 6.1 – Cas d'étude "Lutte contre les feux de forêt"

**Deux visions du collectif** Cette coopération peut être approchée différemment selon qu'on considère un collectif :

— comme une simple conjonction de comportements individuels ;

— comme un tout incarné.

Dans le premier cas, un agent qui participe à un collectif rencontre un problème dans sa propre implication sur une tâche collaborative. Dans la volonté de corriger ce qui devient son problème, il sollicite d'autres agents de son voisinage en ne sachant ni s'ils appartiennent à son propre SMA ni si l'un d'eux a une solution à proposer. Dans notre scénario, l'agent intermédiaire de la communication qui participe au service défaillant et qui a détecté la rupture de la communication va par exemple diffuser un message pour requérir l'assistance d'un tiers pour joindre un des autres agents de la route que devait emprunter initialement le message. La proximité d'un agent drone va lui permettre [4] d'accéder à un point d'entrée dans le collectif SMA2 et de trouver une solution à son problème.

Dans le second cas, la requête de correction du lien de communication est remontée par l'agent qui détecte la panne vers son niveau collectif (SMA1). Si le SMA1 peut observer le produit collectif d'un autre SMA, s'il peut juger de sa pertinence pour répondre à son besoin et s'il peut le solliciter, alors le service de communication pourra être rétabli. Ainsi, le SMA1 aura assuré la résilience d'une compétence qu'il avait déjà (l'acheminement d'un message d'un capteur à la station de collecte). Ce

4. En faisant l'hypothèse des compatibilités techniques (technologies sans fil, protocole d'interaction...).

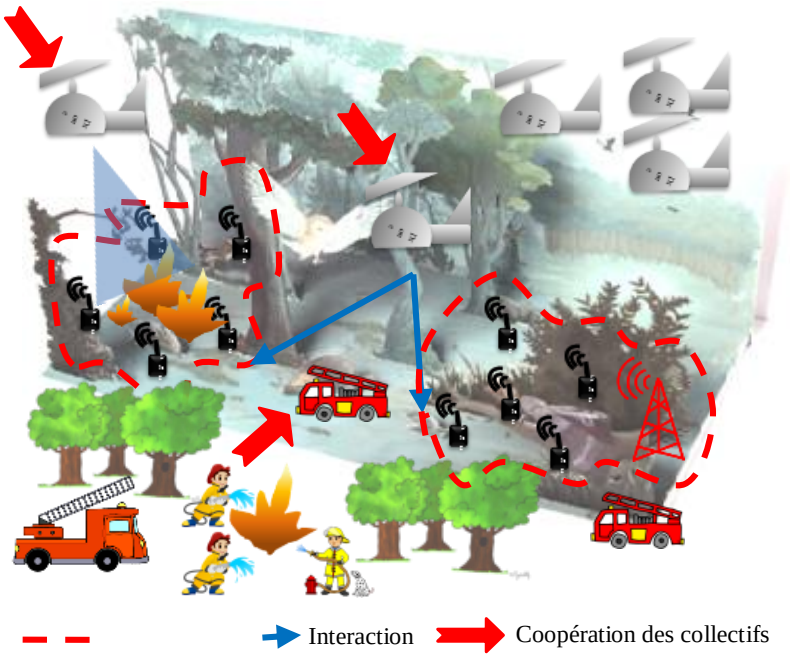



FIGURE 6.2 – Scénario de coopération des collectifs

faisant il peut d'ailleurs acquérir une compétence "extinction de feu".

**L'état d'esprit des agents** Dans la mesure où les SMA sont indépendants, le tempérament que le concepteur du système a insufflé aux agents va fortement impacter la relation de son système aux autres collectifs.

Lorsqu'un conflit apparaitra, un agent de tempérament égoïste privilégiera à tout prix son propre intérêt supérieur (celui du SMA auquel il appartient). Il visera la résilience de son système et développera des stratégies pour veiller à disposer des ressources et des compétences qu'il nécessite au moment opportun.

Un agent coopérant c'est-à-dire qui adopte un tempérament qui consiste à travailler collectivement afin de répondre à des objectifs partagés ou du moins acceptables par tous, privilégiera la coordination et la négociation afin de trouver un compromis.

### 6.3.2 De nouveaux défis

Dans ce contexte, il nous apparait important de relever deux défis liés aux intentions : l'*intersubjectvité* qui fait ici référence à l'intégration de l'intention de l'autre dans un processus de décision individuel et l'*identité collective* qui fait référence à une intention collective.

**L'intersubjectivité** La notion d'intersubjectivité (voir {Andreewsky, 2004}) exportée dans un contexte multi-agent fait référence à la capacité des agents à prendre en considération la "pensée" des autres agents dans leurs propres "jugements". Dans la plupart des applications industrielles des systèmes multi-agents, les agents n'interagissent qu'avec des agents d'un même système (même s'ils défendent des intérêts différents). Aussi, leurs négociations se déroulent dans un cadre précis

qui a été défini par le concepteur du système global. Ce cadre est l'expression de sa perception du collectif. Dans le contexte de nos travaux, et notamment celui du WoT, un agent peut participer à la réalisation de plusieurs missions liées à des applications différentes. Ces applications sont manipulées par plusieurs utilisateurs et/ou organisations qui rendent plus difficile une éventuelle recherche de compromis. Certaines missions, comme celles confiées par le propriétaire de l'objet physique auquel il est associé, sont prioritaires. Lorsqu'un agent s'engage à participer à une tâche collective ou à résoudre un problème collectif, il doit être sûr que cette participation n'empêchera pas la réalisation de ses autres missions. Il ne suffit pas de mesurer le coût du service rendu à l'autre agent mais bien de comprendre l'implication sur les différents objectifs individuels et hérités des collectifs dans lequel il est impliqué. En d'autres termes, l'agent doit être capable d'identifier les intentions des autres afin d'identifier les différentes situations d'interactions qu'il doit maintenir ou qu'il est susceptible de démarrer. Une situation d'interaction est un ensemble de comportements résultant du regroupement d'agents qui doivent agir pour satisfaire leurs objectifs en tenant compte des contraintes provenant des ressources plus ou moins limitées dont ils disposent et de leurs compétences individuelles. Un agent sera par exemple :

- en situation d'indifférence avec les agents qui ont des objectifs compatibles et avec lesquels il n'est pas en compétition pour des ressources ou des compétences ;
  $\Longrightarrow$ L'état d'esprit de l'agent guidera sa participation ou non au collectif.
- en situation de coopération avec d'autres agents qui ont des objectifs compatibles entre eux et qui peuvent s'aider à compenser des manques de compétences ou de connaissances ;
  $\Longrightarrow$ L'état d'esprit de l'agent guidera sa participation ou non au collectif s'il n'est pas dépendant d'autres agents, sinon il coopérera.
- en situation d'antagonisme avec les entités qui ont des buts incompatibles.
  $\Longrightarrow$ Même s'il a un état d'esprit coopérant, l'agent doit être précautionneux !

Une fois la situation d'interaction identifiée, il conviendra de mettre en œuvre les stratégies collectives adéquates. Le succès de ces stratégies permettra de créer et de maintenir des réseaux de confiance qui permettront aux avatars de choisir plus sereinement leurs partenaires.
Ces travaux démarrent avec la thèse de El Mehdi Khalfi débutée en 2014 (financement ANR ASA-WoO).

**L'identité collective** Wittorski {2008} considère la notion d'identité collective comme une intention sociale et souligne que *la constitution d'une identité collective pour un groupe semble répondre d'abord au besoin de se défendre vis-à-vis des contraintes qui lui sont imposées, mais aussi de revendiquer une définition autonome de son propre projet d'existence* (contexte des sciences humaines et sociales). Dans notre cas, elle est liée au problème de l'observation des collectifs et donc plus globalement de l'inscription des phénomènes émergents.

Nous poursuivons sur ce point la partie théorique des travaux sur la modélisation multi-niveau des systèmes multi-agents notamment en approfondissant les pistes prometteuses introduites dans la thèse de T.T.H. Hoang pour une formalisation récursive unificatrice des systèmes multi-agents. Ce travail permet d'incarner les collectifs dans des abstractions qui peuvent être vues comme des individus éthérés. Ces individus permettent notamment de reformuler des problèmes d'interaction *many-to-many* en *one-to-one*. Plus loin, ils permettent d'ailleurs de projeter la notion d'intersubjec-

tivité sur des collectifs.
La thèse de A. Khenifar (projet PHC-Tassili) qui a débuté en 2013 s'inscrit dans ce thème mais sous l'aspect du couplage d'organisations multi-agents plus proche des systèmes de systèmes. Nous proposons dans ce cadre de repenser le couplage de systèmes collectifs en le voyant comme un système d'organisations pour la décision. Il s'agit, dans un premier temps, de proposer une méthode de modélisation à base d'une approche multi-agent pour les systèmes complexes composés de plusieurs sous-systèmes hétérogènes communicants (son cas d'étude est le scénario d'illustration que nous avons utilisé {Khenifar Bessadi et al., 2015}). L'objectif est de concevoir des composants logiciels intelligents légers, qu'il sera possible d'embarquer sur les différents composants du système. Ces composants logiciels permettront aux différents nœuds du système de fonctionner de manière autonome, et de répondre à des requêtes globales du système.

### 6.3.3 Un champ applicatif excitant

Le problème du couplage de plusieurs collectifs et celui de l'articulation de l'individu "objet connecté" dans des CCP fait du Web des Objets un champ applicatif très excitant. L'ensemble des technologies Web permettant de simplifier les aspects opérationnels de l'interopérabilité, il nous permet de nous focaliser sur les aspects purement stratégiques du collectif. Aussi le projet ASAWoO (ANR INFRA, AAP Infrastructures matérielles et logicielles pour la société numérique 2013, 2014-2018) nous permet de développer dans ce contexte plusieurs contributions :

— des méthodes de conception à la fois de type bottom-up et centrées individus (premières réflexions publiées dans {Jamont et al., 2014a}) ;
— des architectures d'agents pour intégrer dans l'avatar les différentes représentations qu'il établit de lui-même, de son environnement et la société d'avatars (première version publiée dans {Mrissa et al., 2015}) ;
— des mécanismes de recherche dans un collectif des compétences ou des connaissances qui manquent aux avatars que ce soit pour répondre à leurs propres besoins individuels ou à des objectifs globaux (première piste publiée dans {Khalfi et al., 2014}).

En ces temps de fusion qui manifestent de la volonté de produire des entités visibles et donc aux dimensions importantes, le risque est grand de normaliser et de rationaliser. De plus en plus souvent, il m'a semblé devoir justifier en différents lieux de l'existence des SMA en tant que spécialité de recherche. Ces dernières perspectives focalisent mon travail sur ce qui me parait être l'essence des SMA : la perception, la définition, la compréhension de collectifs artificiels.

# Bibliographie


S. Abras. *Système domotique Multi-Agents pour la gestion de l'énergie dans l'habitat.* PhD thesis, Université Joseph Fourier, May 2009.

S. Abras, S. Pesty, S. Ploix, and M. Jacomino. Une approche multi-agent pour la gestion de l'énergie dans l'habitat. *Revue d'Intelligence Artificielle*, 24(5) :649–671, 2010.

G. Adams and J.-J. Paques. Gemma, the complementary tool of the grafcet. In *Proceedings of the Fourth Annual Canadian Conference on Programmable Control and Automation Technology Conference and Exhibition*, Toronto, Canada, October 1988. IEEE.

R. Alexander, R. Alexander-Bown, and T. Kelly. Engineering safety-critical complex systems. In *Proceedings of the 1st CoSMoS Workshop*, September 2008.

C. Ambuhl, A. E. Clementi, P. Penna, G. Rossi, and R. Silvestri. Energy consumption in radio networks : Selfish agents and rewarding mechanisms. In *Approximation and Online Algorithms*, volume 2909 of *LNCS*, pages 329–330. Springer, 2004.

E. Andreewsky. De l'individuel au collectif : quelques modèles systémiques. In *Gouvernance individuelle et collective : le point de vue systémique.* AFSET, Juin 2004.

Y. Asnar and P. Giorgini. Risk analysis as part of the requirements engineering. Technical report, University of Trento, Italy, 2007.

S. S. Babu, A. Raha, and M. K. Naskar. Article : Trustworthy route formation algorithm for wsns. *International Journal of Computer Applications*, 27(5) :35–39, August 2011. Published by Foundation of Computer Science, New York, USA.

K. Balasubramanian, J. Balasubramanian, J. Parsons, A. S. Gokhale, and D. C. Schmidt. A platform-independent component modeling language for distributed real-time and embedded systems. *J. Comput. Syst. Sci.*, 73(2) :171–185, 2007.

M. L. Bard. *Architectural Modeling and Analysis of Complex Real-Time Systems.* PhD thesis, Mälardalen University, September 2003.

R. Begg and R. Hassan. Artificial neural networks in smart homes. In *Designing Smart Homes*, volume 4008 of *LNCS*, pages 146–164. Springer, 2006.

C. Bernon, M. P. Gleizes, S. Peyruqueou, and G. Picard. Adelfe : A methodology for adaptive multi-agent systems engineering. In *3rd Int. Workshop on Engineering Societies in the Agents World*, volume 2577, pages 156–169. Springer, 2002.

C. Bernon, V. Camps, M. P. Gleizes, and G. Picard. Adelfe : atelier de développement de logiciels à fonctionnalité émergente. *Technique et Science Informatiques*, 22(4) :387–391, 2003.

D. Blanes, E. Insfran, and S. A. ao. Re4gaia : A requirements modeling approach for the development of multi-agent systems. In *Advances in Software Engineering*, volume 59 of *Communications in Computer and Information Science*, pages 245–252. Springer, 2009. ISBN 978-3-642-10619-4.

B. W. Boehm. A spiral model of software development and enhancement. *IEEE Computer*, 21(5) : 61–72, 1988.

A. Boukerche, L. Xu, and K. El-Khatib. Trust-based security for wireless ad hoc and sensor networks. *Computer Communications*, 30(11-12) :2413–2427, 2007.

M. Bouroche and V. Cahill. We don't need to agree to coordinate. In *Workshop on Dependable Network Computing and Mobile Systems (DNCMS'08) associated with the 27th IEEE Symposium on Reliable Distributed Systems (SRDS 2008)*, pages 47–51, Naples, Italy, October 2008. IEEE.

R. Braatz. Scilab textbook companions [focus on education]. *Control Systems, IEEE*, 34(3) :76–76, June 2014. ISSN 1066-033X.

L. Braubach, A. Pokahr, K.-H. Krempels, and W. Lamersdorf. Deployment of distributed multi-agent systemss. In *Engineering Societies in the Agents World V, ESAW 2004*, volume 3451 of *LNCS*. Springer, 2005.

F. M. T. Brazier, C. M. Jonker, and J. Treur. Principles of component-based design of intelligent agents. *Data Knowledge Engineering*, 41(1) :1–27, 2002.

P. Bresciani, P. Giorgini, H. Mouratidis, and G. Manson. Multi-agent systems and security requirements analysis. In *Software Engineering for Multi-Agent Systems II*, volume 2940 of *LNCS*, pages 352–354. Springer, 2004.

J.-P. Briot. Composants et agents : évolution de la programmation et analyse comparative. *Technique et Science Informatiques*, 33(1-2) :85–115, 2014.

C. Bunse and H.-G. Groß. Unifying hardware and software components for embedded system development. In *Architecting Systems with Trustworthy Components*, volume 3938, pages 120–136. Springer, 2006.

B. D. Carolis, G. Cozzolongo, S. Pizzutilo, and V. L. Plantamura. Agent-based home simulation and control. In *15th International Symposium on Foundations of Intelligent Systems*, volume 3488 of *Lecture Notes in Computer Science*, pages 404–412. Springer, 2005.

A. Carrasco, M. C. Romero-Ternero, F. Sivianes, M. D. Hernández, and J. I. Escudero. Multi-agent and embedded system technologies applied to improve the management of power systems. *JDCTA*, 4(1) :79–85, 2010.

C. Castelluccia and A. Francillon. Protéger les réseaux de capteurs sans fil. In *6e Symposium sur la sécurité des technologies de l'information et des communications*, pages 1–11, 2008.

A. Castor, R. Pinto, C. T. L. L. Silva, and J. Castro. Towards requirement traceability in tropos. In M. Ridao and L. M. Cysneiros, editors, *Workshop em Engenharia de Requisitos*, pages 189–200, 2004.

L. Cernuzzi, M. Cossentino, and F. Zambonelli. Process models for agent-based development. *Journal of Engineering Applications of Artificial Intelligence*, 18 :205–222, 2005.

F. Cervantes, M. Occello, F. Ramos, and J. Jamont. Toward self-adaptive ecosystems of services in dynamic environments. In *Advances in Systems Science - Proceedings of the International Conference on Systems Science 2013, ICSS 2013, Wroclaw, Poland, September 10-12, 2013*, volume 240 of *Advances in Intelligent Systems and Computing*, pages 671–680. Springer, 2014.

A. Chella, M. Cossentino, L. Sabatucci, and V. Seidita. Agile passi : An agile process for designing agents. *International Journal of Computer Systems, Science and Engineering. Special Issue on Software*, 2006.

B. Chen and H. H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *Trans. Intell. Transport. Sys.*, 11(2) :485–497, June 2010. ISSN 1524-9050.

H. Chen, H. Wu, J. Hu, and C. Gao. Agent-based trust management model for wireless sensor networks. In *International Conference on Multimedia and Ubiquitous Engineering*, pages 150–154. IEEE Computer Society, 2008.

H. Chesbrough. Open innovation : The new imperative for creating and profiting from technology. *Harvard Business School Press*, 2006.

J. Choi, S. Oh, and R. Horowitz. Distributed learning and cooperative control for multi-agent systems. *Automatica*, 45(12) :2802–2814, 2009.

G. Conte, D. Scaradozzi, A. Perdon, and G. Morganti. Mas theory for resource management in home automation systems. *Journal of Physical Agents*, 3 :15–19, 2009.

M. Cossentino, L. Sabatucci, and A. Chella. A possible approach to the development of robotic multi-agent systems. In *Proceedings of the 2003 IEEE/WIC International Conference on Intelligent Agent Technology*, pages 539–544. IEEE Computer Society, 2003.

D. Culler, J. Hill, P. Buonadonna, R. Szewczyk, and A. Woo. A network-centric approach to embedded software for tiny devices. In *Proceedings of the First International Workshop on Embedded Software*, volume 2211 of *LNCS*, pages 114–130, London, UK, 2001. Springer. ISBN 3-540-42673-6.

H. Dai, Z. Jia, and Z. Qin. Trust evaluation and dynamic routing decision based on fuzzy theory for manets. *Journal of Sofware*, 4(10) :1091–1101, 2009.

S. A. DeLoach and J. C. García-Ojeda. O-mase : a customisable approach to designing and building complex, adaptive multi-agent systems. *International Journal of Agent-Oriented Software Engineering*, 4(3) :244–280, 2010.

S. A. DeLoach, M. F. Wood, and C. H. Sparkman. Multiagent systems engineering. *International Journal of Software engineering and Knowledge Engineering*, 11(3) :231–258, 2001.

Y. Demazeau. From interactions to collective behaviour in agent-based systems. In *Proceedings of the 1st. European Conference on Cognitive Science*, pages 117–132, Saint-Malo, 1995.

G. Despotou, R. Alexander, and T. Kelly. Addressing challenges of hazard analysis in systems of systems. In *Proceedings of the 3rd IEEE Systems Conference*, March 2009.

B. Douglass. *Real-Time UML : Designing Efficient Objects for Embedded Systems*. Addison-Wesley, 1999.

Y. Du, F. P. Tso, and W. Jia. Architecture design of video transmission between umts and wsn. In *Ubi-Media Computing, 1st IEEE International Conference on*, pages 57–62, 2008. doi : 10.1109/UMEDIA.2008.4570866.

V. Ebrahimipour, K. Rezaie, and S. Shokravi. Enhanced fmea by multi-agent engineering fipa based system to analyze failures. In *Proceedings of Reliability and Maintainability Symposium (RAMS) 2010*, pages 1 – 6, San Jose, CA, USA, January 2010. ISBN 978-1-4244-5102-9.

W. Elmenreich. Intelligent methods for embedded systems. In *Proceedings of the First Workshop on Intelligent Solutions in Embedded Systems*, pages 3–11, 2003.

K. Erciyes, O. Dagdeviren, D. Cokuslu, O. Yilmaz, and H. Gumus. Modeling and simulation tools for mobile ad hoc networks. In *Mobile Ad Hoc Networks : Current Status and Future Trends*, pages 37–70. CRC Press, 2011. ISBN 9781439856505.

I. Fairweather and A. Brumfield. *LabVIEW : A Developer's Guide to Real World Integration*. Taylor and Francis, 2011. ISBN 9781439839812.

P. H. Feiler and D. P. Gluch. *Model-Based Engineering with AADL - An Introduction to the SAE Architecture Analysis and Design Language*. SEI series in software engineering. Addison-Wesley, 2012. ISBN 978-0-321-88894-5.

N. Fournel, M. Minier, and S. Ubéda. Survey and benchmark of stream ciphers for wireless sensor networks. In *First IFIP TC6 / WG 8.8 / WG 11.2 International Workshop on Information Security Theory and Practices*, volume 4462 of *Lecture Notes in Computer Science*, pages 202–214. Springer, 2007.

N. Fourty, A. van den Bossche, and T. Val. An advanced study of energy consumption in an IEEE 802.15.4 based network : Everything but the truth on 802.15.4 node lifetime. *Computer Communications*, 35(14) :1759–1767, 2012.

S. Frenot. Programmation orientée composants. Technical report, INSA de Lyon, December 2002.

F. Fréry, A. Gratacap, and T. Isckia. Les écosystèmes d ?affaires, par-delà la métaphore. *Revue Francaise de Gestion*, 38(222) :69–75, 2012.

A. Galton. Causal reasoning for alert generation in smart homes. In *Designing Smart Homes*, volume 4008 of *LNCS*, pages 57–70. Springer, 2006.

S. Ganeriwal, L. K. Balzano, and M. B. Srivastava. Reputation-based framework for high integrity sensor networks. *ACM Transactions on Sensor Nerworks*, 4(3) :1–37, 2008.

A.-J. Garcia-Sanchez, F. Garcia-Sanchez, J. García-Haro, and F. Losilla. A cross-layer solution for enabling real-time video transmission over ieee 802.15.4 networks. *Multimedia Tools Appl.*, 51(3) : 1069–1104, 2011.

B. Gaudou, A. Herzig, D. Longin, and M. Nickles. A new semantics for the fipa agent communication language based on social attitudes. *Proceedings of the 2006 Conference on ECAI 2006 : 17th European Conference on Artificial Intelligence August 29 – September 1, 2006, Riva Del Garda, Italy*, pages 245–249, 2006.

P. Gautier and L. Gonzalez. *L'Internet des objets : Internet, mais en mieux.* AFNOR, 2011.

L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. Emstar : A software environment for developing and deploying wireless sensor networks. In *Proc. of the USENIX Annual Technical Conference*, pages 283–296. USENIX, June 2004.

M.-P. Gleizes. Vers la résolution de problèmes par émergence, Habilitation à Diriger des Recherches, université paul sabatier, Décembre 2004.

A. Goel and A. Sharma. Performance analysis of mobile ad-hoc network using aodv protocol. *International Journal of Computer Science and Security*, 3(5) :334–343, 2009.

Groupe MARCIA. Auto-organisation := évolution de structure(s). In *PRC-GDR Intelligence artificielle*, pages 139–152. Hermes, 1996.

Groupe MARCIA. L'auto-organisation comme objet d'étude dans les systèmes multi-agents. In *PRC-GDR Intelligence artificielle*, pages 243–260. Hermes, 1997.

M. Guido and C. Montrucchio. Modeling & simulation for experimentation, test & evaluation and training : Alenia aeronautica experiences and perspectives. In *Transforming Training and Experimentation through Modelling and Simulation*, pages 5.1–5.16. NATO Research and Technology Organisation, 2006.

A. Guillemet, G. Haïk, T. Meurisse, J.-P. Briot, and M. Lhuillier. Mise en oeuvre d'une approche componentielle pour la conception d'agents. In *Actes des septièmes journées francophones d'Intelligence Artificielle et systèmes multi-agents*, pages 53–65. Hermès Lavoisier Editions, 1999.

O. Gutiérrez. Multiagent systems interaction through social norms. *Disertation*, 2009.

J. O. Gutiérrez-García, F. F. R. Corchado, and J.-L. Koning. An obligation-based framework for web service composition via agent conversations. *WIAS*, 10(2) :135–150, 2012.

N. Hamani, J.-P. Jamont, M. Occello, and M. Koudil. An optimized self-organized approach to manage communication in wireless instrumentation systems. In *Proceedings of the 2014 IEEE Symposium on Intelligent Agent*, pages 54–61. IEEE, 2011.

G. Han, D. Choi, and W. Lim. A novel sensor node selection method based on trust for wireless sensor networks. In *International Conference on Wireless Communications, Networking and Mobile Computing*, pages 2397 – 2400. IEEE Computer Society, 2007.

M. Handy and D. Timmermann. Simulation of mobile wsn with accurate modelling of non-linear battery effects. In *Applied Simulation and Modelling*. Acta Press, Sep. 2003.

K. Hanninen, J. Maki-Turja, M. Nolin, M. Lindberg, J. Lundback, and K.-L. Lundback. The rubus component model for resource constrained real-time systems. In *Proceedings of the IEEE International Symposium on Industrial Embedded Systems*, pages 177–183. IEEE Industrial Electronics Society, 2008.

S. J. Harmon, S. A. DeLoach, and Robby. Abstract requirement analysis in multiagent system design. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*, WI-IAT '09, pages 86–91. IEEE Computer Society, 2009. ISBN 978-0-7695-3801-3.

S. Hassan, D. Al-Jumeily, and A. Hussain. Autonomic computing paradigm to support system's development. In *Developments in eSystems Engineering (DESE), 2009 Second International Conference on*, pages 273–278, Dec 2009.

X. He, X. Gui, and W. Wei. A heider-theory based reputation framework for wsn. In *10th IEEE International Conference on High Performance Computing and Communications*, pages 635–640. IEEE, 2008.

S. Heath. *Embedded Systems Design*. Butterworth-Heinemann, Newton, MA, USA, 1st edition, 1997. ISBN 0750632372.

T. A. Henzinger and J. Sifakis. The embedded systems design challenge. In *Proceedings of the 14th International Symposium on Formal Methods (FM), Lecture Notes in Computer Science*, pages 1–15. Springer, 2006.

D. E. Herlea, C. M. Jonker, J. Treur, and N. J. E. Wijngaards. Specification of behavioural requirements within compositional MAS design. In *Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, volume 1647, pages 8–27. Springer, 1999.

K. H. S. Hla, Y. Choi, and J. S. Park. Multi agent community to support information processing in wireless sensor network applications. *Int. J. Intell. Inf. Database Syst.*, 4(1) :81–102, Feb. 2010. ISSN 1751-5858.

T. Hoang, M. Occello, and J. Jamont. A generic recursive multiagent model to simplify large scale multi-level systems observation. In *Proceedings of the 2011 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2011, Campus Scientifique de la Doua, Lyon, France, August 22-27, 2011*, pages 155–158. IEEE Computer Society, 2011.

T. Hoang, M. Occello, J. Jamont, and C. B. Yelles. Supervision de systèmes complexes artificiels décentralisés. proposition d'un modèle multi-agent récursif générique. *Revue d'Intelligence Artificielle*, 26(5) :569–600, 2012.

C.-C. Hsu and L.-Z. Wang. A smart home resource management system for multiple inhabitants by agent conceding negotiation. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 607–612, Oct 2008.

Y.-C. Hsu, K. F. Tsai, and J. T. Liu. *Vdhl Modeling for Digital Design Synthesis*. Kluwer Academic Publishers, 1995.

H.-P. Huang, C.-C. Liang, and C.-W. Lin. Construction and soccer dynamics analysis for an integrated multi-agent soccer robot system. In *Natl. Sci. Counc. ROC(A)*, volume 25, pages 84–93, 2001.

R. Huang, Z. Chen, and G. Xu. Energy-aware routing algorithm in wsn using predication-mode. In *International Conference on Communications, Circuits and Systems (ICCCAS)*, pages 103–107, Chengdu, 28-30 July 2010. IEEE.

M. C. Huebscher and J. A. McCann. A survey of autonomic computing&mdash ;degrees, models, and applications. *ACM Comput. Surv.*, 40(3) :7 :1–7 :28, Aug. 2008. ISSN 0360-0300.

G. Hutzler, H. Klaudel, and D. Wang. Towards timed automata and multi-agent systems. In *Formal Approaches to Agent-Based Systems*, volume 3228 of *LNCS*, pages 161–172. Springer, 2005.

C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion : a scalable and robust communication paradigm for sensor networks. In *Proc. of the 6th annual ACM/IEEE Int. conf. on mobile computing and networking*, 2000.

ISO. International standard iso/iec 7498-1, 2nd edition, 1989.

J. Isoré. Coordination et coopération. *Le blog du management*, Mai 2014.

T. Issariyakul and E. Hossain. *Introduction to Network Simulator NS2*. Springer Publishing Company, Incorporated, 3 edition, 2014.

V. Issarny, N. Georgantas, S. Hachem, A. Zarras, P. Vassiliadis, M. Autili, M. A. Gerosa, and A. B. Hamida. Service-oriented middleware for the future internet : state of the art and research directions. *J. Internet Services and Applications*, 2(1) :23–45, 2011. doi : 10.1007/s13174-011-0021-3. URL `http://dx.doi.org/10.1007/s13174-011-0021-3`.

ITU. *ITU Internet Reports 2005 : The Internet of Things*. International Telecommunication Union, 2005.

J.-P. Jamont. *DIAMOND : Une approche pour la conception de systèmes multi-agents embarqués*. PhD thesis, Institut National Polytechnique de Grenoble, September 2005.

J.-P. Jamont and M. Occello. A self-organization process for communication management in embedded multiagent system. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*. IEEE Computer society, 2007.

J.-P. Jamont and M. Occello. Hw/sw cosimulation as a part of wireless intrumentation system design lifecycle : discussion about an experience. In *12th joint Symposium on Man, Sciences and Measurement*, volume 2, pages 289–294. The International Measurement Confederation, 2008.

J.-P. Jamont and M. Occello. Prise en compte des exigences extra-fonctionnelles relatives au déploiement des SMA embarqués. In *Dynamiques, couplages et visions intégratives - JFSMA 13 - Vingt-et-unièmes journées francophones sur les systèmes multi-agents, Lille, France, Juillet 3-5, 2013*, pages 179–188, 2013a.

J.-P. Jamont and M. Occello. Using mash in the context of the design of embedded multiagent system. In *Proc. of the 11th Int. Conference on Practical Applications of Agents and Multiagent Systems*, Advances in Intelligent and Soft Computing. Springer, 2013b.

J.-P. Jamont and M. Occello. Meeting the challenges of decentralized embedded applications using multiagent systems. *International Journal of Agent-Oriented Software Engineering*, pages 1–47, 2016. À paraître.

J.-P. Jamont, M. Occello, and A. Lagreze. A multiagent system for the instrumentation of an underground hydrographic system. In *Proceedings of IEEE International Symposium on Virtual and Intelligent Measurement Systems*, USA, May 2002. IEEE Measurement and Instrumentation Society.

J.-P. Jamont, M. Occello, R. Guillermin, and M. Pezzin. Utilisation de la phase d'analyse de la méthode DIAMOND pour concevoir un système de radiolocalisation. In *Systèmes Multi-Agents, Génie logiciel multi-agents - JFSMA 09 - Dix Septièmes Journées Francophones sur les Systèmes Multi-Agents, Lyon, France, October 19-21, 2009*, pages 175–184, 2009.

J.-P. Jamont, M. Occello, and A. Lagrèze. A multiagent approach to manage communication in wireless instrumentation systems. *Measurement*, 43(4) :489 – 503, 2010. ISSN 0263-2241.

J.-P. Jamont, E. Mendes, and M. Occello. A framework to simulate and support the design of distributed automation and decentralized control systems : Application to control of indoor building comfort. In *IEEE Symposium on Computational Intelligence in Control and Automation*, pages 80–87, Paris, France, 2011. IEEE.

J.-P. Jamont, M. Occello, and E. Mendes. Decentralized intelligent real world embedded systems : a tool to tune design and deployment. In *Proceedings of the 11th International Conference on Practical Applications of Agents and Multiagent Systems*, Advances in Intelligent and Soft Computing, pages 133–144. Springer, 2013.

J.-P. Jamont, L. Médini, and M. Mrissa. A web-based agent-oriented approach to address heterogeneity in cooperative embedded systems. In *Trends in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection (Special Sessions)*, volume 293 of *Advances in Intelligent Systems and Computing*, pages 45–52. Springer, 2014a.

J.-P. Jamont, C. Raïevsky, and M. Occello. Handling safety-related non-functional requirements in embedded multi-agent system design. In *Advances in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection.*, Lecture Notes in Computer Science, pages 159–170. Springer, 2014b.

P. Jaszczyk and D. Król. Updatable multi-agent osgi architecture for smart home system. In *Agent and Multi-Agent Systems : Technologies and Applications*, volume 6071 of *LNCS*, pages 370–379. Springer, 2010.

D. Jawawi, S. Deris, and R. Mamat. Enhancements of pecos embedded real-time component model for autonomous mobile robot application. In *Proceedings of the IEEE International Conference on Computer Systems and Applications*, pages 882–889. IEEE Industrial Electronics Society, 2006.

N. R. Jennings. Foundations of distributed artificial intelligence. chapter Coordination Techniques for Distributed Artificial Intelligence, pages 187–210. John Wiley & Sons, Inc., New York, NY, USA, 1996. ISBN 0-471-006750.

E. Jez. Développement d'une colonie de robots mobiles kurasu. Technical report, Diplome de Recherche Technologique de Grenoble-INP, France, 2011.

C. Johnson. The glasgow-hospital evacuation simulator : Using computer simulations to support a risk-based approach to hospital evacuation. Technical report, University of Glasgow, 2005.

V. Julian and V. Botti. Developing real-time multi-agent systems. *Integr. Comput.-Aided Eng.*, 11 : 135–149, April 2004. ISSN 1069-2509.

E. Karmouch and A. Nayak. A distributed constraint satisfaction problem approach to virtual device composition. *IEEE Trans. Parallel Distrib. Syst.*, 23(11) :1997–2009, 2012.

E. Kendall and M. Malkoun. Design Patterns for the Development of Multi-Agents Systems. In C. Zhang and D. Lukose, editors, *Multi-Agents Systems : Methodologies and Applications, Proceedings of Second Australian Workshop on DAI*, volume LNAI 1286, pages 17–32, Cairns, Australia, august 1997. Springer-Verlag.

J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1) :41–50, Jan 2003.

E. M. Khalfi, J.-P. Jamont, F. Cervantes, and M. Barhamgi. Designing the web of things as a society of autonomous real/virtual hybrid entities. In *Proceedings of the 2014 International Workshop on Web Intelligence and Smart Sensing*, IWWISS '14, pages 16 :1–16 :5, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2747-3.

A. Khenifar Bessadi, J.-P. Jamont, M. Occello, and M. Koudil. Vers une coopération des collectifs de systèmes multi-agents hétérogènes. In *Treizièmes rencontres des jeunes chercheurs en Intelligence Artificielle*, Juillet 2015.

I. Kim, H. Park, B. Noh, Y. Lee, S. Lee, and H. Lee. Design and implementation of context-awareness simulation toolkit for context learning. In *Proceedings of the IEEE Conf. on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pages 96–103, 2006.

H. Kopetz. *Real-Time Systems : Design Principles for Distributed Embedded Applications*. Springer Publishing Company, Incorporated, 2nd edition, 2011. ISBN 1441982361, 9781441982360.

M. Koudil. *Une approche orientée objet pour le codesign*, Thèse d'état, Institut National d'Informatique d'Alger. Juin 2002.

S. Kubicki, Y. Lebrun, S. Lepreux, E. Adam, C. Kolski, and R. Mandiau. Simulation in contexts involving an interactive table and tangible objects. *Simulation Modelling Practice and Theory*, 31 :116–131, 2013.

S. Kumar. *The Codesign of Embedded Systems : An Unified Hardware Software Representation*. Kluwer Academic Publishers, 2002.

V. T. Le, N. Bouraqadi, S. Stinckwich, V. Moraru, and A. Doniec. Making networked robots connectivity-aware. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 1835–1840, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-2788-8.

E. A. Lee. Embedded software. In *Advances in Computers*, page 2002. Academic Press, 2002.

E. A. Lee. Cyber physical systems : Design challenges. In *11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2008), 5-7 May 2008, Orlando, Florida, USA*, pages 363–369, 2008.

S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *Procedings of the IEEE Int. Conf. on Communications*, pages 3201–3205, 2001.

P. Leitao, V. Marik, and P. Vrba. Past, present, and future of industrial agent applications. *Industrial Informatics, IEEE Transactions on*, 9(4) :2360–2372, Nov 2013. ISSN 1551-3203.

N. G. Leveson. A new accident model for engineering safer systems. *Safety Science*, 42(4) :237–270, april 2004.

P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim : accurate and scalable simulation of entire tinyos application. In *Proc. of the Int. Conf. on Embedded Networked Sensor Systems*, pages 126–137. ACM, 2003.

N. Lewis, N. Foukia, and D. G. Govan. Using trust for key distribution and route selection in wireless sensor networks. In *IEEE/IFIP Network Operations and Management Symposium : Pervasive Management for Ubioquitous Networks and Services*, pages 787–790. IEEE, 2008.

J. Li, R. Conradi, C. Bunse, M. Torchiano, O. P. N. Slyngstad, and M. Morisio. Development with off-the-shelf components : 10 facts. *IEEE Software*, 26(2) :80–87, 2009.

J. Lind. *Iterative Software Engineering for multiagent systems : The MASSIVE Method*, volume 1994. Springer, 2001.

A. N. Lindoso and R. Girardi. The sramo techique for analysis and reuse of requirements in multi-agent application engineering. In *WER*, pages 41–50, 2006.

T. Liu and M.-Z. Zhou. Implementing a trust-aware routing protocol in wireless sensor nodes. In *IEEE International Conference on Progress in Informatics and Computing (PIC)*, pages 556 – 559. IEEE Computer Society, 2010.

Y. Lu, G. Zhao, and F. Su. Adaptive ant-based dynamic routing algorithm. *In Proceedings of the 5th World Congress on Intelligent Control and Automation*, pages 2694–2697, 2004.

J. Luo, L. Xu, J.-P. Jamont, L. Zeng, and Z. Shi. Flood decision support system on agent grid : method and implementation. *Enterprise Information Systems*, 1(1) :49–68, 2007.

B. Ma. A novel stereoscopic security architecture with trust management for wireless sensor networks. In *Proceedings of the 2009 International Conference on Communication Software and Networks*, ICCSN '09, pages 797–800, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3522-7.

Z. Maamar, S. K. Mostéfaoui, and H. Yahyaoui. Toward an agent-based and context-oriented approach for ws composition. *IEEE Trans. Knowl. Data Eng.*, 17(5) :686–697, 2005.

T. W. Malone and K. Crowston. What is coordination theory and how can it help design cooperative work systems ? In *CSCW '90, Proceedings of the Conference on Computer Supported Cooperative Work, Los Angeles, CA, USA, October 7-10, 1990*, pages 357–370. ACM, 1990.

T. W. Malone and K. Crowston. The interdisciplinary study of coordination. *ACM Comput. Surv.*, 26(1) :87–119, 1994.

M. K. Marina and S. R. Das. Ad hoc on-demand multipath distance vector routing. *Mobile Computing and Communications Review*, 6(3) :92–93, 2002.

G. Martin, B. Bailey, and A. Piziali. *ESL Design and Verification : A Prescription for Electronic System Level Methodology*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007. ISBN 9780080488837.

L. V. Massawe, F. Aghdasi, and J. Kinyua. The development of a multi-agent based middleware for rfid asset management system using the passi methodology. *Information Technology : New Generations, Third International Conference on*, 0 :1042–1048, 2009.

M. Mendes, B. Santos, and J. S. da Costa. A matlab/simulink multi-agent toolkit for distributed networked fault tolerant control systems. In *Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, 2010.

Y. Meng. An agent-based reconfigurable system-on-chip architecture for real-time systems. *2nd Int. Conference on Embedded Software and Systems*, 0 :166–173, 2005.

A. Mercier, M. Occello, and J.-P. Jamont. Using multiagent self-organization techniques for seeking information in virtual social communities. *Web Intelligence and Agent Systems*, 10(2) :235–246, 2012.

A. Mercier, C. Raïevsky, M. Occello, and D. Genthial. Solutions multi-agents pour la prise en charge à domicile des séniors. *Ingénierie des Systèmes d'Information*, 18(6) :83–112, 2013a.

A. Mercier, C. Raïevsky, M. Occello, and D. Genthial. Solutions multi-agents pour la prise en charge à domicile des séniors. *Ingénierie des Systèmes d'Information*, 18(6) :83–112, 2013b.

T. Meurisse and J. Briot. Une approche á base de composants pour la conception d'agents. *Revue Technique et Science Informatiques, Numéro Spécial "Réutilisation"*, 20(4) :583–602, avril 2001.

L. Monostori, J. Vancza, and S. Kumara. Agent-based systems for manufacturing. *PCIRP Annals - Manufacturing Technology*, 55(2) :697–720, 2006.

H. Moore. *MATLAB for engineers*. ESource–the Prentice Hall engineering source. Prentice Hall, 20011. ISBN 9780136044222.

A. J. Morais. A multi-agent approach for web adaptation. In *7th International Conference on Practical Applications of Agents and Multi-Agent Systems*, volume 55 of *Advances in Intelligent and Soft Computing*, pages 349–355. Springer, 2009.

S. Moreno and E. Peulot. *Le GEMMA : Modes de marches et d'arrêts, GRAFCET de coordination des tâches, conception des systèmes automatisés de production sûrs : réglementation ...* Casteilla, 2009. ISBN 978-2713531286.

M. Morge. Système dialectique au travers duquel les agents argumentatifs jouent et arbitrent : vers une prise de décision collective et débattue. In *Systèmes Multi-Agents, vers la conception de systèmes artificiels socio-mimétiques, JFSMA 05, treizièmes journées francophones sur les systèmes multi-agents, Calais, France, November 23-25, 2005*, pages 115–127. Hermes Lavoisier Editions, 2005.

M. Mrissa, L. Médini, and J. Jamont. Semantic discovery and invocation of functionalities for the web of things. In *2014 IEEE 23rd International WETICE Conference, WETICE 2014, Parma, Italy, 23-25 June, 2014*, pages 281–286. IEEE, 2014.

M. Mrissa, L. Médini, J. Jamont, N. L. Sommer, and J. Laplace. An avatar architecture for the web of things. *IEEE Internet Computing*, 19(2) :30–38, 2015.

J. P. Müller and K. Fischer. Application impact of multi-agent systems and technologies : A survey. In O. Shehory and A. Sturm, editors, *Agent-Oriented Software Engineering : Reflections on Architectures, Methodologies, Languages, and Frameworks*, pages 27–53. Springer, 2014. ISBN 978-3-642-54431-6.

S. Munroe, T. Miller, R. Belecheanu, M. Pechoucek, P. McBurney, and M. Luck. Crossing the agent technology chasm : Lessons, experiences and challenges in commercial applications of agents. *Knowledge Eng. Review*, 21(4) :345–392, 2006.

A. Myles, D. B. Johnson, and C. E. Perkins. A mobile host protocol supporting route optimization and authentication. *IEEE Journal on Selected Areas in Communications*, 13(5) :839–849, 1995.

H. R. Naji, B. E. Wells, and L. Etzkorn. Creating an adaptive embedded system by applying multi-agent techniques to reconfigurable hardware. *Future Gener. Comput. Syst.*, 20 :1055–1081, August 2004. ISSN 0167-739X.

T. V. Nguyen, H. A. Nguyen, and D. Choi. Development of a context aware virtual smart home simulator. *CoRR*, 1007.1274, 2010.

T. Nishiyama and T. Sawaragi. A decision-making modeling for home ambient intelligent system and its extension to multi-agent modeling. In *System Integration (SII), 2011 IEEE/SICE International Symposium on*, pages 354–357, Dec 2011.

M. Occello. MIRAGE : a generic model to build recursive multiagent systems. In *International Conference on Artificial Intelligence*, pages 649–655, Las Vegas, USA, 2000. CSREA.

M. Occello and Y. Demazeau. Une approche du temps réel dans la conception d'agents. In J. Muller and J. Quinqueton, editors, *4 èmes Journées Francophones IAD-SMA*, pages 101–112, Port Camargue, France, Avril 1996. Hermès.

M. Occello, Y. Demazeau, and C. Baeijs. Designing organized agents for cooperation in a real time context. In *Collective Robotics*, volume LNCS/LNAI 1456, pages 25–73. Springer-Verlag, March 1998.

M. Occello, J. Koning, A. Ibriz, and M. Erradi. A multi-formalism method for designing and validating intelligent agent-based systems. In *2002 International Conference on Intelligent Information Technology Proceedings - ICIT 2002*, pages 53–59. People's Post and Telecommunications Publishing House, September 2002.

M. Occello, J.-P. Jamont, R. Guillermin, and M. Pezzin. A multiagent approach for an uwb location embedded software architecture. In *Proceedings of the 5th Int. Conf. on Soft Computing as Transdisciplinary Science and Technology*, pages 279–285. ACM, 2008.

B. C. Occello M., Koning J. L. Conception de systèmes multi-agents : quelques éléments de réflexion méthodologique. *Technique et science informatiques*, 20(2) :233–263, 2001. ISSN 0752-4072.

J. Olascuaga-Cabrera, E. Lopez-Mellado, A. Mendez-Vazquez, and F. F. Ramos-Corchado. A self-organization algorithm for robust networking of wireless devices. *Sensors Journal, IEEE*, 11(3) : 771–780, 2011.

J. G. Olascuaga-Cabrera, A. Mendez-Vazquez, and E. Lopez-Mellado. A novel distributed energy-efficient self-organized algorithm for wireless ad hoc networks. In *Intelligent Environments (IE), 2012 8th International Conference on*, pages 19–26. IEEE, 2012.

E. O'Neill, M. Klepal, D. Lewis, T. O'Donnell, D. O'Sullivan, and D. Pesch. A testbed for evaluating human interaction with ubiquitous computing environments. In *Proceedings of the 1st Int. Conf. on Testbeds, Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, pages 60–69. IEEE Computer Society, 2005.

S. Park, A. Savvides, and M. B. Srivastava. Simulating networks of wireless sensors. In *Proceedings of the 2001 Winter Simulation Conference*, pages 1330–1338. ACM, December 2001.

M. Pechoucek and V. Marík. Industrial deployment of multi-agent technologies : review and selected case studies. *Autonomous Agents and Multi-Agent Systems*, 17 :397–431, 2008. ISSN 1387-2532.

Q. Pei, L. Wang, H. Yin, L. Pang, and H. Tang. Layer key management scheme on wireless sensor networks. In *Proceedings of the Fifth International Conference on Information Assurance and Security*, pages 427–431. IEEE Computer Society, 2009.

X. Peng and Z. Wei. The trustworthiness based on hash chain in wireless sensor network. In *IEEE/IPIP International Conference on Embedded and Ubiquitous Computing*, pages 101–106. IEEE Computer Society, 2008.

C. E. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. In *IETF Internet draf draft-ietf-manet-aodv-08.txt*. IETF, 2003.

V. T. Pham, C. Raïevsky, and J. Jamont. A multiagent approach using model-based predictive control for an irrigation canal. In *Trends in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection, 12th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2014) Special Sessions, Salamanca, Spain, June 4-6, 2014.*, pages 87–94, 2014.

G. Picard and M.-P. Gleizes. The adelfe methodology - designing adaptive cooperative multi-agent systems (chapitre 8). In *Methodologies and Software Engineering for Agent Systems : The AOSE handbook*, pages 157–176. Kluwer Publishing, 2004.

R. D. Pietro, P. Michiardi, and R. Molva. Confidentiality and integrity for data aggregation in wsn using peer monitoring. *Security and Communication Networks*, 2(2) :181–194, 2009.

G. Polaków and M. Metzger. Agent-based framework for distributed real-time simulation of dynamical systems. In *Proc. of the 3rd KES Int. Symp. on Agent and Multi-Agent Systems : Technologies and Applications*, pages 213–222, Berlin, 2009. Springer. ISBN 978-3-642-01664-6.

F. Ponci, A. Deshmukh, A. Monti, L. Cristaldi, and R. Ottoboni. Interface for multi-agent platform systems. In *Proceedings of the IEEE Instrumentation and Measurement Technology Conference*, pages 2226 – 2230. IEEE, 2005.

G. J. Pottie and W. J. Kaiser. *Principles of Embedded Networked Systems Design.* Cambridge University Press, 2009. ISBN 9780521095235.

E. R. Lakner an Nemeth, K. M. Hangos, and I. Cameron. Agent-based diagnosis for granulation processes. In *9th Symposium on Process Systems Engineering and the 16th European Symposium on Computer Aided Process Engineering*, pages 1443–1448, 2006.

D. Raggett. The web of things : Extending the web into the real world. In *SOFSEM 2010 : Theory and Practice of Computer Science, 36th Conference on Current Trends in Theory and Practice of Computer Science, Spindleruv Mlýn, Czech Republic, January 23-29, 2010. Proceedings*, pages 96–107, 2010.

C. Raievsky, J.-P. Jamont, L. Lefevre, et al. Irrigation canals distributed model-based predictive control using multi-agent systems. In *MOSIM 2014, 10ème Conférence Francophone de Modélisation, Optimisation et Simulation*, 2014.

A. Raja, M. Barley, and X. Zhang. Towards safe coordination in multi-agent systems. In *Safety and Security in Multiagent Systems*, volume 4324 of *LNCS*, pages 1–7. Springer, 2009.

L. Razo, F. Ramos, and M. Occello. Metaose : Meta-analysis for agent oriented software engineering. In *Electronics, Robotics and Automotive Mechanics Conference (CERMA), 2010*, pages 197–204, Sept 2010.

A. Ricci, L. Tummolini, M. Piunti, O. Boissier, and C. Castelfranchi. Mirror worlds as agent societies situated in mixed reality environments. In *Proocedings of the 17th International Workshop on Coordination, Organisations, Institutions and Norms*, Lecture Notes in Computer Science. Springer, 2014.

D. Richards, S. Splunter, F. Brazier, and M. Sabou. Composing web services using an agent factory. In *Extending Web Services Technologies*, volume 13 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 229–251. Springer US. ISBN 978-0-387-23343-7.

C. R. Robinson, P. Mendham, and T. Clarke. A multiagent approach to manage communication in wis. *Journal of Physical Agents*, 4(3) :489 – 503, 2010. ISSN 0263-2241.

C. Rodriguez-Fernández and J. J. Gómez-Sanz. Self-management capability requirements with selfmml & ingenias to attain self-organising behaviours. In *Proc. of the 2nd Int. Workshop on Self-organizing Architectures*, pages 11–20. ACM, 2010. ISBN 978-1-4503-0087-2.

P. Roques and F. Vallée. *UML en action.* Eyrolles, 2003.

S. Russel and P. Norvig. *Artificial Intelligence : a Modern Approach.* Prentice-Hall, 1995.

J. Sabater-Mir and L. Vercouter. Trust and reputation in multiagent systems. In G. Weiss, editor, *Multi-Agent Systems (2nd edition)*, chapter 9. MIT Press, 2012.

H. Salzwedel. Comparison of can, flexray, and ethernet architectures for the design of abs systems. *SAE Technical Paper*, 2011.

O. Schira. Conception d'une couche physique adaptative pour réseau de terrain en vue de l'instrumentation de réseaux hydrographiques souterrains,. In *Mémoire de DEA Signal Image Parole Télécom , Institut National Polytechnique de Grenoble*, 2003.

E. Shakshuki and H. Malik. Agent based approach to minimize energy consumption for border nodes in wireless sensor network. In *Proc. of the 21st Int. Conf. on Advanced Networking and Applications*, pages 134–141, USA, 2007. IEEE. ISBN 0-7695-2846-5.

M. P. Singh. Agent communication languages : Rethinking the principles. *IEEE Computer*, 31(12) : 40–47, 1998. doi : 10.1109/2.735849. URL `http://doi.ieeecomputersociety.org/10.1109/2.735849`.

M. P. Singh. A social semantics for agent communication languages. *Issues in Agent Communication*, pages 31–45, 2000.

J. Skazinski. Automation revolutionizes embedded systems diagnostics. *Embedded Computing Design*, winter :28–29, 2003.

A. Sobieh and J. Hou. A simulation framework for sensor networks in j-sim, November 2003.

N. I. Spanoudakis and P. Moraitis. An ambient intelligence application integrating agent and service-oriented technologies. In *SGAI Conf.*, pages 393–398, 2007.

L. Sterling and K. Taveter. *The Art of Agent-Oriented Modelling*. MIT Press, 2009.

D. B. Stewart, R. Volpe, and P. K. Khosla. Design of dynamically reconfigurable real-time software using port-based objects. *IEEE Transaction on Software Engineering*, 23(12) :759–776, 1997.

M. Takimoto, M. Mizuno, M. Kurio, and Y. Kambayashi. Saving energy consumption of multi-robots using higher-order mobile agents. In *Agent and Multi-Agent Systems : Technologies and Applications*, volume 4496 of *LNCS*, pages 549–558. Springer, 2007.

J. Teich. Hardware/software codesign : The past, the present, and predicting the future. *Proceedings of the IEEE*, 100(Special Centennial Issue) :1411–1430, May 2012. ISSN 0018-9219.

G. Tesauro, D. M. Chess, W. E. Walsh, R. Das, A. Segal, I. Whalley, J. O. Kephart, and S. R. White. A multi-agent systems approach to autonomic computing. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '04, pages 464–471, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 1-58113-864-4.

J. Thangarajah, G. B. Jayatilleke, and L. Padgham. Scenarios for system requirements traceability and testing. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 285–292. IFAAMAS, 2011.

L. Thévin, F. Badeig, J. Dugdale, O. Boissier, and C. Garbay. Un système multi-agent normatif pour la collaboration et l'interaction mixte. In *Principe de Parcimonie - JFSMA 14 - Vingt-deuxièmes Journées Francophones sur les Systèmes Multi-Agents, Loriol-sur-Drôme, France, Octobre 8-10, 2014*, pages 203–212, 2014.

R. Thomas. Colonie d'objets semi-vivants : Conception d'un système multi-agent embarqué. Technical report, Diplome de Recherche Technologique de Grenoble-INP, France, 2010.

B. Titzer, D. K. Lee, and J. Palsberg. Avrora : scalable sensor network simulation with precise timing. In *Proceedings of the 4th Int. Symp. on Information Processing in Sensor Networks*, pages 477–482. IEEE, 2005.

F. Vahid and T. Givargis. *Embedded System Design : A Unified Hardware/Software Introduction.* John Wiley & Sons, 2002. ISBN 0471386782.

H. Van Dyke Parunak. A practitioners ? review of industrial agent applications. *Autonomous Agents and Multi-Agent Systems*, 3(4) :389–407, 2000.

R. C. van Ommering, F. van der Linden, J. Kramer, and J. Magee. The koala component model for consumer electronics software. *IEEE Computer*, 33(3) :78–85, 2000.

Venture Development Corp. *Embedded Systems Bulletin.* Venture Development Corp., 2003.

L. Vercouter. Mast : Un modèle de composant pour la conception de sma. In *Première Journée Multi-Agent et Composant*, Paris, novembre 2004.

L. Vercouter and J.-P. Jamont. Lightweight trusted routing for wireless sensor networks. In *Proceedings of the 9th International Conference on Practical Applications of Agents and Multi-Agent Systems, Advances in Intelligent and Soft-Computing*, volume 2, pages 501–504. Lecture Notes in Artificial Intelligence, Springer, March 2011.

L. Vercouter and J.-P. Jamont. Lightweight trusted routing for wireless sensor networks. *Progress in Artificial Intelligence*, 1(2) :193–202, 2012.

W. Cesário et al. Object-based hardware/software component interconnection model for interface design in system-on-a-chip circuits. *Journal of Systems and Software archive*, 70(3) :229–244, 2004.

L. Wang, Y. Shu, M. Dong, L. Zhang, and O. W. W. Yang. Adaptive multipath source routing in ad hoc networks. In *IEEE Int. Conf. on Communications*, volume 3, pages 867–871, 2001.

V. Werneck, A. Y. Kano, and L. M. Cysneiros. Evaluating adelfe methodology in the requirements identification. In *10th Workshop in Requirements Engineering*, pages 13–24, 2007.

D. Weyns, K. Schelfthout, T. Holvoet, and T. Lefever. Decentralized control of e'gv transportation systems. In *4rd Int. Joint Conf. on Autonomous Agents and Multiagent Systems - Industrial Applications*, pages 67–74. ACM, 2005.

E. White. *Making Embedded Systems : Design Patterns for Great Software.* O'Reilly Media, Inc., 2011. ISBN 1449302149, 9781449302146.

R. Wittorski. La notion d'identité collective. In *La question identitaire dans le travail et la formation : contributions de la recherche, état des pratiques et étude bibliographique.* Harmattan, 2008.

M. Wooldridge, N. Jennings, and D. Kinny. The gaia methodology for agent-oriented analysis and design. In *Autonomous Agents and Multi-Agent Systems*, volume 3, pages 285–312. Kluwer Academic Publishers, 2000.

C.-L. Wu, C.-F. Liao, and L.-C. Fu. Service-oriented smart-home architecture based on osgi and mobile-agent technology. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 37(2) : 193–205, 2007.

L. Yang, L. Zetao, L. Yi, and Z. Wei. A novel energy-efficient data gathering algorithm for wireless sensor networks. In *IEEE Proceedings of the 8th World Congress on Intelligent Control and Automation*, Jinan, China, July 6-9 2010.

C. Yanrong and Q. Hang. A novel rumor routing for wireless sensor network. In *Fourth International Conference on Genetic and Evolutionary Computing*, 13-15 Dec 2010.

F. Zambonelli, N. R. Jennings, and M. Wooldridge. Developing multiagent systems : The gaia methodology. *ACM Transaction on Software Engineering and Methodology*, 12(3) :317–370, 2003.

E. Zelkha, B. Epstein, S. Birrell, and C. Dodsworth. From devices to ämbient intelligence¨ :the transformation of consumer electronics. Technical report, Philips, June 1998.

Y. Zhang, L. D. Kuhn, and M. P. J. Fromherz. Improvements on ant routing for sensor networks. In *4th International Workshop on Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *Lecture Notes in Computer Science*, pages 154–165. Springer, 2004.

X. Zhu. Pheromone based energy aware directed diffusion algorithm for wireless sensor network. In *Proc. of the 3rd Int. Conf. on Intelligent Computing*, volume 4681 of *LNCS*, pages 283–291. Springer, 2007.

R. Zurawski. Networked embedded systems : An overview. In *Networked Embedded Systems*. CRC Press, 2009. ISBN 9781439807552.

# Acronymes

***MWAC** Famille des modèles basés sur MWAC. 52

**AADL** Architecture Analysis and Design Language. 36

**AAII** Australian AI Institute. 30

**ACLIRSYS** Advanced Control for Low Inerty Refrigeration System. 105

**ACO** Ant Colony Optimization. 65, 66, 68, 69

**ADELFE** Atelier de DEveloppement de Logiciels à Fonctionnalité Emergente. 22

**ADEME** Agence de l'Environnement et de la Maîtrise de l'Energie. 106

**ADR** Adaptive ant-based Dynamic Routing. 67

**AEIO** Agent, Environnement, Interaction, Organisation. 5, 31, 47

**ANR** Agence Nationale de la Recherche. 5, 49, 82, 86, 105, 106, 110, 111

**AntMWAC** Ant based Multi-Wireless-Agent Communication model. vii, 53, 54, 64, 65, 69, 70, 104

**AoC** Agent on Chip. 39, 105

**AODV** Ad-hoc On-demand Distance Vector. 52, 67

**AP** Access Point. 51

**API** Application Programming Interface. 49, 92

**ASAWoO** Adaptive Supervision of Avatar/Object Links for the Web of Objects. 49, 82, 86, 105, 110, 111

**ASCML** Agent Society Configuration Manager and Launcher. 19

**ASTRO** Agent Spécialisé Temps-Réel Organisé. 5, 47

**AUML** Agent Unified Modelling Langage. 30, 31, 33

**BDI** Belief, Desire, Intention. 36

**CCP** Collectif Cyber-Physique. vii, ix, 4, 6, 7, 10, 12, 13, 15–18, 20, 22, 25, 36, 43, 44, 47, 51–56, 58, 60, 64, 65, 70, 71, 73, 76, 77, 81–83, 85, 89, 94, 102–104, 106, 107, 111

**CEA** Commissariat à l'Énergie Atomique et aux énergies alternatives. 21, 33, 88, 93, 105

**COSY** Cooperative Complex Systems. 1, 6

**COTS** Commercial Off-The-Shelf. 35

**CPU** Central Processing Unit. 55, 84

**DEA** Diplôme d'Études Approfondies. 85

**DESIRE** DEsign and Specification of Interacting REasoning components. 30

**DIAMOND** Decentralized Iterative Approach for Multiagent Open Networks Design. vii, 4, 6, 17, 20, 24, 30, 31, 37, 44, 45, 89, 93, 102

**DoS** Deny of Service. 59

**DRT** Diplôme de Recherche Technologique. 21, 93, 94

**DSDV** Destination-Sequenced Distance Vector. 56, 58

**DSR** Dynamic Source Routing. 56–58

**eASTRO** embedded ASTRO. 5, 47, 48, 81, 104

**ENF** Exigence Non Fonctionnelle. 22, 24, 25

**ENVSYS** ENVironment SYStem. 54, 84, 105

**EPLD** Erasable Programmable Logic Device. 8

**FITT** Fond Incitatif de Transfert de Technologie. 21, 48, 54, 84, 85

**FMEA** Failure mode and effects analysis. 23

**FPGA** Field-Programmable Gate Array. 8, 93, 106

**FSM** Finite State Machine. 31

**GDR** Groupe De Recherche. 28

**GEMMA** Guide d'Etude des Modes de Marche et d'Arrêt. 25

**GPS** Global Positioning System. 92

**GRAFCET** GRAphe Fonctionnel de Commande des Étapes et Transitions. 25

**HAZOP** HAZard and OPerability study. 23

**HDL** Hardware Description Langage. 31, 45

**HTTP** Hypertext Transfer Protocol. 50

**I³** Information - Interaction - Intelligence. 28

**IEEE** Institute of Electrical and Electronics Engineers. 1, 52

**IETF** Internet Engineering Task Force. 1

**IHM** Interface Homme-Machine. 34

**IP** Internet Protocol. 2

**IRISA** Institut de Recherche en Informatique et Systèmes Aléatoires. 49

**IRIT** Institut de Recherche en Informatique de Toulouse. 28, 106

**ISO** International Standart Organization. 27

**ITU** International Telecommunication Union. 2

**IUT** Institut Universitaire de Technologie. 1

**LCIS** Laboratoire de Conception et d'Intégration des Systèmes. 1, 106

**LETI** Laboratoire d'Électronique et de Technologie de l'Information. 21, 33, 105

**LHC** Laboratoire Hubert Curien. 105

**LIG** Laboratoire d'Informatique de Grenoble. 22

**LIRIS** Laboratoire d'InfoRmatique en Image et Systèmes d'information. 49, 105

**LITEN** Laboratoire d'Innovation pour les Technologies des Énergies Nouvelles et les Nanomatériaux. 21, 88

**LITIS** Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes. 106

**LMCS** Laboratoire de Méthodes de Conception de Systèmes. 21, 105

**M2M** Machine-to-machine. 2

**MAC** Medium Access Control. 52, 53

**MANET** Mobile Ad hoc NETwork. 1

**MASC** MultiAgent for Supply Chaining. 28

**MASC** MultiAgent System Codesign tool. 36–38

**MASE** MultiAgent Systems Engineering methodology. 19, 30

**MASH** MultiAgent Software and Hardware simulator. vii, 6, 85–91, 93, 94, 96, 98–102, 104, 106

**MASoC** MultiAgent on Chip. 39, 105

**MASSIVE** MultiAgent SystemS Iterative View Engineering. 19

**MESSAGE** Methodology for engineering systems of software agents. 29, 30

**METALISM** Meta-Analyse pour l'ingénierie des systèmes multi-agents. 105

**MPSoC** Multi-Processor System on Chip. 39, 105

**MWAC** Multi-Wireless-Agent Communication model. vii, 5, 52–58, 62, 64, 65, 67–71, 76, 77, 92, 104

**OS** Operating System. 48

**OSI** Open Systems Interconnection. 52, 53, 73, 74, 78

**OTS** Off-The-Shelf. 35

**PAL** Programmable Array Logic. 8

**PASSI** Process for Agent Societies Specification and Implementation. 30

**PBO** Port-Based Object. 35

**PDU** Protocol Data Unit. 73

**PECOS** PErvasive COmponent Systems. 35

**PHC** Projet Hubert Curien. 48, 105

**PICML** Plateform-Independant Component Modeling Langage. 35

**PKI** Public Key Infrastructure. 60

**POO** Programmation Orientée Objet. 11

**QG** Quartier Général. 107

**QoS** Quality of Service. 52, 66

**RAM** Random Access Memory. 8

**REST** Representational State Transfer. 50

**RFID** Radio Frequency Identification. 14, 39, 49, 86

**ROM** Read Only Memory. 8

**RPC** Remote Procedure Call. 11

**RT** Real Time. 37

**RT-MESSAGE** Real Time - Methodology for engineering systems of software agents. 29

**SCP** Système cyber-physique. 15, 105

**SCXML** State Chart XML. 50

**SER** Système Embarqué en Réseau. 10, 11, 15

**SIET** Solution Intelligente, Ergonomique et Tactile. 21, 105

**SMA** Système Multi-Agent. vii, ix, 4–6, 9, 12–17, 19, 20, 22–33, 36, 39, 43–45, 47, 51, 55, 56, 71–73, 77, 79, 81–84, 86, 87, 89, 90, 92, 96, 97, 103, 105, 107–109, 111

**SMA-R** Système Multi-Agent Réccursif. 71–74

**SMAC** Systèmes Multi-Agents Coopératifs. 28

**SOA** Service Oriented Architecture. 78, 79

**SoC** System on Chip. 105

**SWANS** Silicon platforms for Wireless Advanced Networks of Sensors. 21

**SyCoMas** Systèmes Complexes et Multiagents. 39

**TrustedMWAC** Trusted Multi-Wireless-Agent Communication model. vii, 53, 54, 58, 61–63, 104

**ULB** Ultra Large Bande. 37

**UML** Unified Modelling Langage. 30, 31, 37

**UPMF** Université Pierre Mendès France. 1

**VHDL** Very high speed integrated circuit Hardware Description. 41

**WoT** Web of Things. 2, 49, 50, 82, 110

**XML** Extensible Markup Language. vii, 98, 99

# Index

## Démarche, modèles et outils multi-agents pour l'ingénierie des collectifs cyber-physiques

**Résumé :** Nous appelons *collectif cyber-physique* un système embarqué en réseau dans lequel les nœuds ont une autonomie de décision et coopèrent spontanément afin de participer à l'accomplissement d'objectifs du système global ou de pallier des manques de connaissances ou de compétences individuelles. Ces objectifs portent notamment sur l'état de leur environnement physique.

La conception de ces collectifs présente de nombreux défis. Ce mémoire d'Habilitation propose une discussion des différents aspects de l'ingénierie de ces systèmes que nous modélisons en utilisant le paradigme multi-agent.
Tout d'abord, une méthode complète d'analyse et de conception est proposée. Ses différentes particularités sont discutées au regard des différents défis précédemment évoqués. Des modèles d'agent et de collectifs adaptés aux communications contraintes et aux environnements changeants sont alors proposés. Ils permettent de simplifier la conception des collectifs cyber-physiques. Enfin, un outil qui permet la simulation et le déploiement de systèmes collectifs mixtes logiciels/matériels est introduit.

Ces contributions ont été éprouvées dans des projets académiques et industriels dont les retours d'expériences sont exploités dans les différentes discussions.

**Mots-clés :** Système Multi-Agent ; Collectif Cyber-Physique ; Analyse Orientée Agent ; Architecture d'Agent ; Application réelle.

## Multi-Agent Approach, Models and Tools to Collective Cyber-Physical System Engineering

**Abstract :** We call a Collective Cyber-Physical System (CCPS), a system consisting of numerous autonomous execution units achieving tasks of control, communication, data processing or acquisition. These nodes are autonomous in decision making and they can cooperate to overcome gaps of knowledge or individual skills in goal achievement.

There are many challenges in the design of these collective systems. This Habilitation thesis discusses various aspects of such a system engineering modeled according to a multi-agent approach.

First, a complete CCPS design method is proposed. Its special features are discussed regarding the challenges mentioned above. Agent models and collective models suitable to constrained communications and changing environments are then proposed to facilitate the design of CCPS. Finally, a tool that enables the simulation and the deployment of hw/sw mixed collective systems is presented.

These contributions have been used in several academic and industrial projects whose experience feedbacks are discussed.

**Keywords :** Multi-agent System ; Collective Cyber-Physical System ; Agent Oriented Analysis ; Agent Design ; Real World Application.