Optimisation des flux logistiques : vers une gestion avancée de la situation de crise

Aida Kaddoussi

N° d'ordre : 209

ECOLE CENTRALE DE LILLE

THESE

Présentée en vue
d'obtenir le grade de

DOCTEUR

En

Spécialité : Automatique, Génie Informatique, Traitement du Signal et Image

Par

Aida KADDOUSSI

DOCTORAT DELIVRE PAR L'ECOLE CENTRALE DE LILLE

Optimisation des flux logistiques: vers une gestion avancée de la situation de crise

Soutenue le 26 Novembre 2012 devant le jury d'examen :

Président	*Etienne, CRAYE, Professeur, Ecole Centrale de Lille*
Rapporteur	*Jacques, CARLIER, Professeur, Université de Technologie de Compiègne*
Rapporteur	*Alain, QUILLIOT, Professeur, Université Blaise Pascal de Clermont -Ferrand*
Membre	*Patrick, SIARRY , Professeur, Université Paris-Est Créteil Val-de-Marne*
Membre	*Jean-Charles, DECONNINCK, Président Generix Group*
Membre	*Francis, BRETAUDEAU, Directeur du Département Logistique, CASSIDIAN*
Directeur de thèse	*Slim, HAMMADI, Professeur, Ecole Centrale de Lille*
Co-encadreur	*Hayfa, ZGAYA, MCU, ILIS - Université Lille2*

Thèse préparée dans le Laboratoire LAGIS UMR CNRS 8219 à l'Ecole Centrale de Lille

Ecole Doctorale SPI 072
PRES Université Lille Nord-de-France

Dédicace

A mon très cher papa, Ridha, pour sa patience, ses précieux conseils, les valeurs et principes qu'il m'a inculqués, et tous ses sacrifices,

A ma douce maman, Souad, pour tout son amour, sa tendresse et son affection sans limites,

A ma très chère sœur Sarra, pour tout ce qu'elle a fait pour moi et à qui je dois tout,

A mon futur époux, Ameur, pour son amour, sa présence et son soutien sans faille,

A ma très chère Karama, pour tous les bons moments passés ensemble,

A la mémoire de mes deux petites sœurs parties trop tôt, Entissar et Tharoua,

A tous ceux et celles qui me sont chers, qui ont participé de près ou de loin à l'aboutissement de ce travail,

Je vous rends hommage par ce modeste travail en guise de ma reconnaissance éternelle et de mon incommensurable amour.

Remerciements

Je tiens à exprimer en premier lieu ma gratitude à mon directeur de thèse Monsieur Slim HAMMADI, professeur à l'Ecole Centrale de Lille et directeur de cette thèse. Je le remercie pour ses conseils judicieux, la grande confiance qu'il m'a accordée et pour la vision innovante de la recherche qu'il a toujours su m'inculquer.

Mes sincères remerciements vont aussi au Professeur Étienne CRAYE, Professeur à l'École Centrale de Lille et directeur de cet honorable établissement, pour l'honneur qu'il m'a fait en acceptant de présider ce jury.

J'adresse aussi mes vifs remerciements au Professeur Jacques CARLIER, Professeur à l'Université de Technologie de Compiègne et au Professeur Alain QUILLIOT, Professeur à l'Université Blaise Pascal de Clermont-Ferrand, qui m'ont fait le grand honneur d'accepter de rapporter cette thèse. Je les remercie infiniment pour le temps consacré à cet effet en dépit de toutes les responsabilités qu'ils ont.

Je souhaite remercier tout particulièrement Messieurs Patrick SIARRY, Professeur, Université Paris-Est Créteil Val-de-Marne, Jean-Charles DECONNINCK, Président Generix Group et Francis BRETAUDEAU, Directeur du Département Logistique à EADS CASSIDIAN, pour avoir évalué ce travail et accepté de participer au jury de cette thèse.

Je souhaite exprimer également toute ma gratitude à Mademoiselle Hayfa ZGAYA, maître de conférences à la faculté d'Ingénierie et Management de la Santé à l'Université Lille 2 et Co-encadrante de cette thèse, pour l'énorme soutien scientifique et pour sa perspicacité qui ont contribué à la réalisation de ce travail. Je la remercie pour sa présence et ses encouragements sans faille.

Mes remerciements vont bien évidemment à l'ensemble du personnel du LAGIS, et plus particulièrement à mes voisines de bureau *Wided* et *Safa*, et à mon amie de toujours *Karama*, grâce à qui j'ai rédigé mon mémoire dans une atmosphère à la fois studieuse et chaleureuse.

Enfin, une pensée très particulière et une reconnaissance infinie à ma grande sœur *Sarra*, sans qui, ce travail n'aurait pas pu arriver à terme. Je lui dois tout.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Les travaux de thèse présentés dans ce mémoire, ont été effectués dans la perspective d'apporter une solution pour des problèmes de prise de décision rencontrés dans des systèmes d'information pour les chaînes logistiques. La chaîne logistique représente un avantage concurrentiel que les entreprises cherchent à perpétuer. Elle a pour but d'optimiser les échanges, ou flux, que l'entreprise entretient avec ses fournisseurs et ses clients. Ces flux peuvent être de natures diverses. Il peut s'agir de flux d'informations, relatifs aux approvisionnements ou à la conception des produits, de flux financiers liés aux achats, ou encore de flux de marchandises (à partir des matières premières et pièces d'assemblage jusqu'aux produits finis).

En particulier, la logistique de gestion de crise fait de plus en plus parler d'elle. Pour ces systèmes, où il n'est jamais très aisé d'anticiper l'évolution de l'environnement, les formes de changements subis sont variées et rapides. En effet, les prises de décision des logisticiens concernent des actions qui ont lieu dans un environnement sans cesse mouvementé. De plus, répondant à un appel d'urgence, elles n'ont, par définition, aucune manière de prévoir ce qui va arriver. Une chaîne logistique de gestion de crise peut à tout moment être victime de retards de livraison, de mauvaises estimations de consommation, de pertes de cargaisons, de pics spontanés de consommation, et bien d'autres évènements imprévisibles. Tous ces imprévus sont susceptibles d'engendrer des ruptures de stocks à n'importe quel point de la chaîne logistique, ce qui peut avoir des conséquences dramatiques pouvant entraîner des pertes humaines. Ces situations extrêmes n'étant pas acceptables, ceci justifie le besoin de créer un outil qui permettrait de simuler des situations logistiques réelles et/ou probables, l'objectif étant d'observer les comportements des différentes zones en place et dégager les meilleures stratégies à adopter selon les situations de crise.

Les travaux de recherche présentés dans cette thèse, dans le cadre d'une coopération avec le département logistique de CASSIDIAN EADS, visent à apporter une réponse à ces défis, dans une démarche qui lie les méthodes d'optimisation au paradigme de l'intelligence artificielle. Nous nous proposons de ce fait de trouver des modèles mathématiques, et des protocoles de coopération inter-agents, permettant de minimiser le risque de rupture de stock dans une zone quelconque de la chaîne logistique.

Afin de mettre en évidence toutes les contraintes qui nous sont posées, nous proposons la conception et le développement d'un système d'aide à la décision orienté agents, pour la modélisation et l'optimisation de la chaîne logistique de gestion de crise. En effet, aspirant à l'élaboration d'un système opérationnel, performant et compétitif à grande échelle, nous nous sommes focalisés sur la notion d'optimisation en usant des méthodologies d'intelligence artificielle distribuée. Les systèmes multi-agents (SMA) sont particulièrement sollicités dans la mise en œuvre de telles applications en raison de leur adaptation pour la représentation comportementale des entités qui composent le système et pour l'étude dynamique de leurs interactions.

Les problèmes de gestion des flux abordés sont envisagés aussi bien au niveau global entre plusieurs acteurs de la chaîne, qu'au sein des activités d'un des acteurs. Ainsi des problèmes d'optimisation allant du positionnement des zones logistiques de gestion de crise à ceux de la prise de décision au sein de chaque zone font partie de nos problématiques abordées.

Le manuscrit se décompose selon les chapitres suivants :

1. Dans le premier chapitre, nous présentons le contexte de recherche qui relève de la gestion des chaînes logistiques et nous mettons l'accent sur la présentation des caractéristiques de la logistique distribuée avancée et de ses problématiques. A travers ce premier chapitre, nous explicitons les choix ayant orienté nos travaux vers la conception et le développement d'un système multi-agent ;
2. Notre problématique ainsi définie, nous nous focalisons dans le deuxième chapitre sur la méthodologie de résolution à adopter et qui se présente sous forme d'une alliance des systèmes multi-agents et des fondements de l'optimisation, pour la mise en place d'une approche s'inscrivant dans le cadre d'une intelligence artificielle distribuée ;
3. Dans le troisième chapitre nous étudions l'apport des systèmes multi-agents dans la conception et la réalisation des systèmes logistiques. Nous proposons ensuite une organisation multi-agent dédiée à la modélisation et l'optimisation d'une chaîne logistique de gestion de crise en détaillant l'architecture de notre système et les comportements des différents agents;
4. Le chapitre 4 sera consacré aux solutions proposées pour optimiser les flux de la chaîne logistique de gestion de crise. En effet, nous proposons dans ce chapitre, un module pour l'optimisation du positionnement géographique des différentes zones de la chaîne, et une approche innovante pour l'ordonnancement distribué des tâches de livraison qui permet d'apporter une solution à un problème de gestion des flux fortement distribué ;

5. Enfin, nous détaillons dans le dernier chapitre, les applications du système adopté pour démontrer la validité des solutions proposées.

Chapitre I : Logistique distribuée avancée

I.1 Introduction

Dans le cadre de nos travaux, nous nous sommes intéressés à l'étude de l'optimisation de la chaîne logistique de gestion de crise et notamment à l'optimisation des activités de planification et de pilotage des flux logistiques, en tenant compte des contraintes de type coût, délai et qualité de service.

Dans ce premier chapitre, nous définissons en premier abord les notions de logistique et de gestion des chaînes logistiques. Ensuite nous nous intéressons aux spécificités de certaines chaînes logistiques, notamment dans le domaine du transport, de la santé et de la gestion de crise. Il s'en suit une présentation des forces et faiblesses des différentes approches existantes en termes de modélisation et pilotage des chaînes logistiques. Ceci va nous permettre, de définir d'un côté les besoins méthodologiques spécifiques à la gestion des chaînes logistiques, et d'un autre, d'argumenter notre motivation pour adopter le courant de recherche qui s'intéresse à la décentralisation de la prise de décision logistique.

Ensuite nous présentons le partenaire industriel avec qui nous menons nos travaux, ce qui nous permet de situer le besoin industriel et de formuler la proposition de recherche que nous apportons pour couvrir ce besoin.

I.2 Logistique et gestion des chaînes logistiques

I.2.1 Logistique - Définitions

On cite souvent la définition d'origine militaire : « La logistique consiste à apporter ce qu'il faut, là où il faut et quand il le faut. »

Le mot « logistique » apparaît en France au XVIIIe siècle, avec l'apparition des problèmes de soutien militaire (réapprovisionnement en armes, munitions, vivres, …).

Ce terme s'est ensuite répandu, dans le milieu industriel notamment, pour évoquer principalement la manutention et le transport des marchandises. Jusqu'aux années 70, la logistique n'avait que peu d'importance dans la gestion des entreprises, considérée comme une fonction secondaire, limitée aux tâches d'exécution dans des entrepôts et sur les quais d'expédition. Mais la logistique est ensuite comprise comme un lien opérationnel entre les différentes activités de l'entreprise, assurant la cohérence et la fiabilité des flux-matières, en

vue de la qualité du service aux clients tout en permettant l'optimisation des ressources et la réduction des coûts.
La logistique devient, au milieu des années 90, une fonction globalisée voire mondialisée de gestion du flux physique dans une vision complète de la chaîne Clients/Fournisseurs, et constitue véritablement une nouvelle discipline du management des entreprises. La « logistique globale » représente ainsi l'ensemble des activités internes ou externes à l'entreprise qui apportent de la valeur ajoutée aux produits et des services aux clients (Courty, 2003).

Dans leur ouvrage, « La logistique au service de l'entreprise », (Colin, Mathé, & Tixié, 1981) ont proposé la définition suivante :

« *La logistique est le processus stratégique par lequel l'entreprise organise et soutient son activité. A ce titre, on peut déterminer et gérer les flux matériels et informationnels afférents, tant internes qu'externes, en amont qu'en aval.* »

La fonction logistique désignerait ainsi la gestion des flux physiques de matières premières et de produits ainsi que celle des flux d'information, c'est à dire les transports, les entrepôts, l'informatique, etc.

Il existe plusieurs types de logistiques :
- logistique d'approvisionnement qui permet d'alimenter les stocks des entreprises et usines en matières premières, composants et sous-ensembles nécessaires à la production.
- logistique de production qui consiste à rendre disponibles les matériaux et les composants nécessaires à la production au pied des lignes de production.
- logistique de distribution qui consiste à acheminer vers le client final ou le consommateur les produits dont il a besoin.
- logistique militaire qui a pour objectif de transporter sur un théâtre d'opérations les forces et les ressources nécessaires pour assurer leur mise en œuvre opérationnelle et maintenir leur soutien.
- rétro-logistique qui consiste à reprendre des produits dont le client ne veut pas ou qu'il veut faire réparer, ou encore des produits à traiter en déchets industriels.

Il y a donc bien des logistiques différentes jusqu'à ce que le concept de *supply chain* ne vienne apporter une certaine unité en ce domaine.

I.2.2 Le concept de la *supply chain*

I.2.2.1 Supply Chain SC

C'est un concept relativement récent même si les militaires utilisent la même expression depuis beaucoup plus longtemps. On définit assez souvent la *supply chain* comme « La suite des étapes de production et de distribution d'un produit depuis les fournisseurs des fournisseurs des producteurs, jusqu'aux clients de ses clients » (Supply Chain Council).

Afin de mieux comprendre le concept de Chaîne Logistique CL (*Supply Chain* SC), nous proposons d'effectuer une revue des définitions de ce terme, utilisées dans la littérature. Christopher (Christopher, 1992) définit la chaîne logistique comme étant «*le réseau d'entreprises qui participent, en amont et en aval, aux différents processus et activités qui créent de la valeur sous forme de produits et de services apportés au consommateur final. En d'autres termes, une chaîne logistique est composée de plusieurs entreprises, en amont (fourniture de matières et composants) et en aval (distribution), et du client final* ».

Lummus (Lummus et al., 1998) a, quant à lui, définit la chaîne logistique comme étant «*le réseau d'entités par lequel le flux matériel passe. Ces entités incluent fournisseurs, transporteurs, sites d'assemblages, centres de distribution, détaillants et clients* ».

Une définition plus générale est celle proposée par Poirier (Poirier et Reiter, 2001) : *« Une chaîne logistique est le système grâce auquel les entreprises amènent leurs produits et leurs services jusqu'à leurs clients ».*

I.2.2.2 Supply Chain Management SCM

Il existe une distinction entre la « chaîne logistique » et la « gestion de la chaîne logistique ». En effet, la gestion de chaîne logistique regroupe les approches, processus et fonctions indispensables pour la réduction des coûts d'une chaîne logistique et l'augmentation de sa flexibilité en vue d'optimiser sa performance.

Ici encore, on relève plusieurs définitions de la gestion de la chaîne logistique (Mentzer, et al., 2001). Beaucoup d'auteurs soulignent la difficulté de définir le *SCM*. Voici quelques définitions, issues notamment de (Croom et al., 2000) :

Vakharia (Vakharia, 2002) définit la SCM comme étant «l'art et la science de créer et d'accentuer les rapports synergiques entre les partenaires d'une même chaîne logistique ayant comme objectif commun de livrer, juste à temps, les bons produits et les bons services au bon client, avec la meilleure quantité ».

Simchi-Levi propose dans son ouvrage la définition suivante (Simchi-Levi et Kaminsky, 2003) « Le *SCM* est une stratégie qui vise à la fois la réduction des frais globaux, permettant

une position plus concurrentielle à toutes les différentes parties de la chaîne logistique, et l'optimisation de la satisfaction du client final par une plus grande adaptabilité des systèmes de production et de distribution ».

Pour Rota-Franz (Rota-Franz et al., 2001), faire du *SCM* consiste à intégrer l'ensemble des moyens internes et externes pour répondre à la demande des clients. L'objectif est d'optimiser de manière simultanée et non plus séquentielle l'ensemble des processus logistiques.

Le principal objectif du *SCM* est d'améliorer la compétitivité industrielle en : 1) minimisant les coûts, 2) assurant le niveau de service requis par le client, 3) allouant efficacement les activités sur les acteurs de production, distribution, transport et d'information ; veillant à ce que les acteurs ne développent pas de comportements locaux antagonistes venant affecter la performance globale.

I.2.3 La prise de décision en logistique : un processus hiérarchisé

Pour n'importe quel type de chaîne logistique, la prise de décision est divisée en trois niveaux: stratégique, tactique et opérationnel, correspondant respectivement à des horizons à long, moyen et court terme, comme l'illustre la figure I.1. Quelques problématiques et travaux rattachés à chacun des niveaux décisionnels sont présentés ci-dessous (Ganeshan et al., 1998, Shapiro, 1999, Vincent et al., 2004, Botta-Genoulaz, 2005).

I.2.3.1 Niveau stratégique

Ce niveau, aussi appelé *Strategic Management* par (Croom et al., 2000) ou encore *Strategic Planning* par (Thomas et Griffin, 1996), regroupe toutes les décisions stratégiques. Ces décisions sont des directives et des lignes d'actions sur le long terme (de 6 mois à plusieurs années), comme, par exemple, la recherche de nouveaux partenaires industriels, la sélection des fournisseurs et sous-traitants, mais aussi les décisions d'implantation ou de délocalisation de zones d'intervention dans le cas de la logistique militaire, l'affectation d'une nouvelle zone d'approvisionnement à un centre de distribution (entrepôt), le développement d'un nouveau produit, la configuration de la chaîne logistique, son mode de fonctionnement, ainsi que les objectifs financiers à atteindre.

I.2.3.2 Niveau tactique

Le niveau décisionnel tactique s'intéresse aux décisions à moyen terme (de quelques semaines à quelques mois) qui devront être exécutées pour déployer la stratégie décidée par l'entreprise. Ces décisions portent sur les problèmes liés à la gestion des ressources de

l'entreprise, en particulier la planification des activités en tenant compte des ressources disponibles sur un horizon fixé.

I.2.3.3 Niveau opérationnel

En ce qui concerne le niveau opérationnel, ou *Operational Planning* selon (Thomas et Griffin, 1996), les décisions ont une portée plus limitée dans l'espace et dans le temps (décisions sur la journée ou sur la semaine). A ce niveau, les décisions tactiques génèrent un plan détaillé de production ou d'ordonnancement, applicable au niveau d'un atelier ou d'une zone logistique.

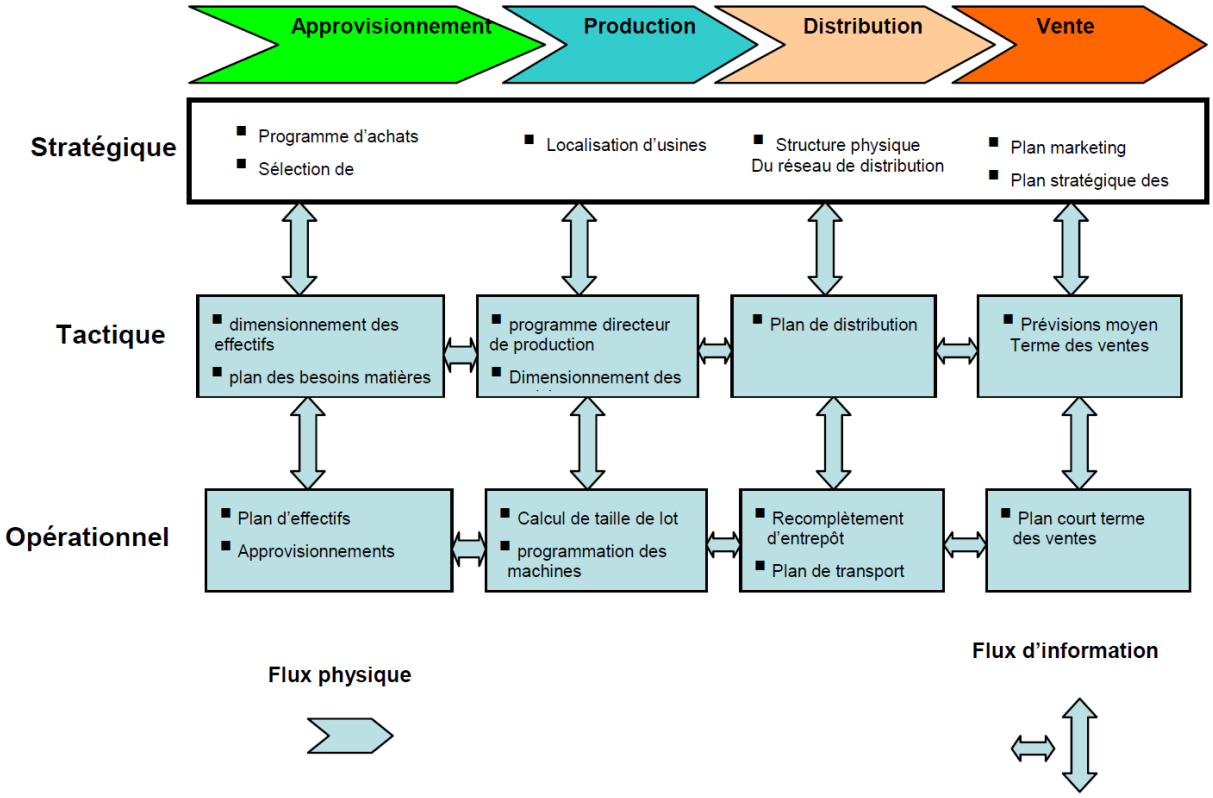

Figure I. 1 Niveaux stratégique, tactique et opérationnel dans une SC (Rhode et al., 00)

I.2.4 Les performances

La gestion de la chaîne logistique cherche à améliorer le système global de production. Pour atteindre cet objectif, nous avons souvent recours à un certain nombre d'indicateurs de performance. Ces indicateurs, parfois difficiles à quantifier, peuvent être la satisfaction du client, le respect des délais de livraison, la flexibilité de la chaîne, le partage de l'information, la gestion des risques, l'amélioration de la traçabilité, etc. Ils sont construits à partir du suivi des stocks et permettent de fixer les seuils des objectifs à atteindre.

Trois principaux indicateurs de performance de la chaîne logistique sont largement utilisés, correspondant chacun à un type de flux : des indicateurs de « coopération » en ce qui

concerne la performance du flux d'information, les coûts pour le flux financier et les délais de livraison pour le flux physique.

La première étape du suivi des performances consiste donc à « mesurer la performance ». Plusieurs critères de performance sont envisageables. Beamon (Beamon, 1998) classe celles-ci en deux catégories : les mesures de performance qualitatives (satisfaction du client, flexibilité, intégration du flux physique et d'information, gestion du risque financier, etc.) et quantitatives (retards de livraison, temps de réponse client, etc.). Ensuite, il faut prendre des décisions de réingénierie et agir sur le système et le modèle à travers des variables de décision afin de tendre vers les objectifs fixés, comme le montre la figure I.2. La mise en place d'un système performant traduit donc un besoin de contrôle de la chaîne logistique et d'amélioration des performances.

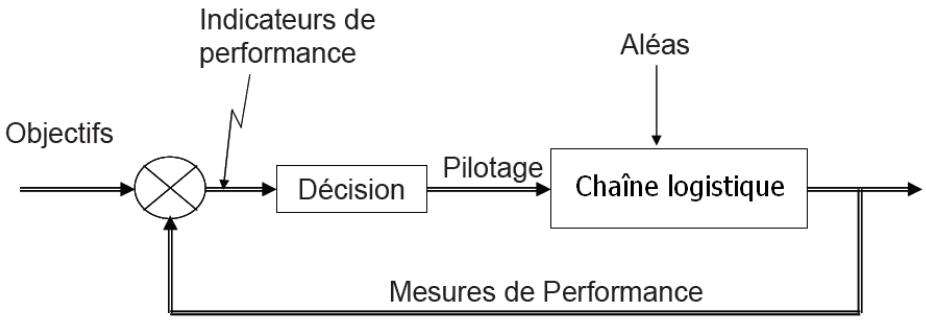

Figure I. 2 Système de contrôle pour les performances des CL

La gestion des systèmes logistiques a pour objectif de développer des modèles et des méthodes d'optimisation permettant de fournir une aide à la décision efficace. En effet, les systèmes logistiques en général constituent des organisations sociotechniques particulièrement difficiles et présentent des problématiques de modélisation et d'optimisation complexes. Les systèmes logistiques concernés sont principalement ceux qui relèvent de : la Production, le Transport, la Santé et la Gestion de Crise. De tels systèmes sont souvent dynamiques, distribués et étendus sur des réseaux à grandes échelles et se présentent généralement sous formes d'entités autonomes en interaction. Les processus issus de ces systèmes sont complexes, par leurs dimensions importantes (nombre très important de variables), la nature de leurs relations dynamiques, et la multiplicité des contraintes auxquelles ils sont soumis (contraintes de productivité et de sécurité pour l'homme et l'environnement).

Les logisticiens de ces systèmes sont confrontés à des problèmes de complexité croissante comme par exemple : Comment améliorer, sécuriser, et optimiser les flux logistiques?

Comment améliorer la synchronisation des flux dans ces systèmes distribués, tels que : les réseaux de soins, la chaîne logistique globale du transport multimodal, les réseaux de production multi-site et la gestion de crise multizone ? Quelles nouvelles technologies de l'information et de la communication adopter et comment les implanter en harmonie avec les spécificités de l'activité de ces systèmes logistiques ?

Ces travaux de recherche qui s'inscrivent dans le cadre des thèmes de recherche de l'équipe Optimisation des Systèmes Logistiques (OSL-LAGIS), se veulent une réponse à ces défis, dans une démarche qui allient les méthodes d'optimisation au paradigme multi-agent..

Le reste de ce chapitre présente les principaux systèmes logistiques ainsi que les technologies existantes pour améliorer leur gestion.

I.3 Logistique des transports

La logistique des transports couvre deux grandes catégories : les systèmes de transport des personnes, en rapport avec la gestion des villes et l'aménagement urbain, et les systèmes de transport de biens, appelés également la logistique de distribution.

I.3.1 Transport des personnes

Dans le domaine du transport des personnes, nous mettons en relation principalement trois intervenants différents. D'un côté, les transports : l'infrastructure (réseau de transport) et équipements (Bus, tram, train, etc.), de l'autre : les exploitants qui investissent et gèrent les réseaux de transport. Enfin, il faut prendre en compte les voyageurs qui utilisent le réseau de transport dans leurs déplacements. Le transport des personnes peut alors être traité selon deux volets d'investigation : l'aide à la mobilité coté client, et l'aide à la régulation coté régulateur. Nous introduisons tout d'abord la notion de transport multimodal comme une solution qui permet de fournir plus de flexibilité et de souplesse dans les déplacements, rendant ainsi le réseau de transport plus fiable et beaucoup plus rentable.

1.3.1.1 Transport multimodal

Un réseau de transport multimodal, dans le cas des systèmes de transport des personnes, est caractérisé par la présence simultanée de différents modes de transport tels que les bus, les métros et les tramways. La liaison entre ces modes est établie au niveau de pôles d'échange ou de nœuds de correspondance. Par rapport au cas monomodal, la planification du trafic dans un réseau multimodal est plus difficile, étant donnée la complexité des déplacements. En effet, ce processus nécessite l'ordonnancement des véhicules, afin de leur affecter des courses et aussi l'ordonnancement du personnel, afin de leur affecter des services.

Sachant que l'information dans le transport est un élément clé, il faut faire un choix entre les multiples systèmes d'information destinés à exploiter au mieux le réseau (Feki, 2010).

1.3.1.2 Systèmes d'informations côté régulateur

Ces Systèmes d'Informations (SI) sont aussi appelés systèmes d'informations côté exploitant. En effet, l'exploitation d'un réseau de transport passe principalement par deux phases-clé : une phase de planification et une phase de régulation. On trouve donc des Systèmes d'Aide à l'Exploitation (SAE) et des Systèmes d'Aide à la Décision (SAD) utilisés pour la réalisation de ces deux phases.

La première phase est la phase de planification réalisée en amont de la mise en service du réseau. Elle consiste à concevoir, planifier et ordonnancer les ressources du transport. C'est une phase importante pour les exploitants ; elle est valable pour les véhicules et pour la planification de la main-d'œuvre : les conducteurs (Freling, 1999). La problématique de la planification est directement liée aux algorithmes d'ordonnancement et particulièrement aux algorithmes de tournées de véhicules (*Vehicle Scheduling Problem VSP*) (Daduna et al, 1995). Actuellement, plusieurs entreprises de transport reposent toutes leurs phases de planification sur des outils, logiciels et systèmes de planification pour générer les tableaux horaires et les fiches de services des conducteurs (Rousseau, 1985).

La deuxième phase survient suite à la mise en place du service de transport. C'est une phase de régulation qui consiste à affiner les horaires et le nombre de véhicules aux heures de pointe pour améliorer la qualité de service et optimiser l'utilisation du réseau.

1.3.1.3 Systèmes d'informations côté client

Le deuxième type de SI lie le client (le voyageur) et les transports. Ce sont les Systèmes d'Informations d'Aide au Déplacement (SIAD). L'objectif principal de ce type de SI est d'aider le voyageur dans la phase de planification en proposant le chemin le plus court ou le moins coûteux. Actuellement, les systèmes d'informations destinés aux clients sont généralement monomodaux et dans le cas où ces systèmes sont multimodaux, ils ne concernent qu'un seul opérateur et donc sont mono-opérateur. Ces systèmes se présentent sous forme de site web. Ils offrent en plus des informations usuelles (d'horaires, de disposition des stations dans une carte) un calculateur d'itinéraire interne. Il s'agit d'un moteur de recherche d'itinéraire accédant aux données locales d'un seul exploitant. Il fournit donc des itinéraires monomodaux. Parmi les exploitants proposant de tels sites nous pouvons citer Transpole pour la métropole Lilloise et la RATP pour l'Ile de France. Tous ces systèmes demeurent mono-opérateur. De plus en plus de projets sont mis en place dans le but d'intégrer les données

issues de plusieurs opérateurs. Les travaux réalisés ont suivi principalement deux stratégies, la première vise à centraliser les données de tous les opérateurs dans un énorme gisement de données et l'exploiter par la suite, la deuxième tend à exploiter les données distantes en créant un système médiateur entre les systèmes existants.

I.3.2 Transport de biens

Le transport de biens couvre les domaines fonctionnels de la logistique de distribution et d'approvisionnement. Ça consiste globalement à mettre à la disposition du client les produits de qualité, en quantité et dans les délais promis et à un coût économique et concurrentiel. La logistique de distribution est une fonction transversale. Sa définition comporte :

- le choix du mode de transport : route, air, voie navigable, combiné,
- le choix entre le transport propre, externalisé, hybride,
- le type d'emballage, normes,
- la gestion du stockage, de la manutention, la localisation des dépôts,
- le service après-vente.

Cette activité recouvre les trafics effectués par la route, le rail, les voies d'eau et la mer, l'air ainsi que les oléoducs. La route représente la part la plus importante des transports terrestres.

I.3.2.1 Les participants

- Le transporteur en compte propre ou « transporteur privé »

Il s'agit généralement d'une entreprise ou d'un particulier qui transporte ses propres marchandises avec ses propres véhicules ou des véhicules loués et avec le plus souvent ses propres conducteurs. Ces transporteurs privés s'opposent aux transporteurs publics.

- Le transporteur public

Il s'agit de ce qu'on appelle messagerie, messagerie rapide, mono-colis, transport de lots industriels, etc. On distingue entre l'envoi de colis à des particuliers, l'envoi de colis rapide par des entreprises à d'autres entreprises pour des colis n'excédant pas les 31 kg, et la messagerie rapide qui permet d'envoyer des colis sur palettes d'un poids maximal allant jusqu'à 3 tonnes.

- L'expéditeur ou « chargeur »

C'est celui qui fait appel à un transporteur. Il est souvent plus reconnu sous l'appellation de donneur d'ordre.

- Le destinataire

Le destinataire est un participant important au transport routier. Il possède des obligations qui concernent le déchargement des marchandises, les conditions de réception des marchandises et les possibilités de réclamation en cas de problèmes.

- Les véhicules du transport routier

Il existe plusieurs types de véhicules : camionnette « fourgon », 6x4 plateau, 6x4 benne, porteur, semi-remorque, camion remorque ou train routier. Des progiciels d'aide au chargement permettent d'optimiser la disposition des colis, palettes et matériels divers à l'intérieur du véhicule en question, compte tenu de l'ordre des livraisons d'une tournée, de la facilité de déchargement et du poids maximal.

I.3.2.2 Paramètres de configuration du réseau de distribution

La définition de l'architecture physique d'un réseau de distribution doit prendre en compte un nombre important de paramètres, notamment :

- caractéristiques, volume, variété et valeur des produits ;

- nombre, localisation, stabilité des points de livraison ;

- niveau de service fixé en termes de délais, fiabilité des livraisons, réactivité aux aléas, de traçabilité ;

- impact des conditionnements sur les types de moyens de transport utilisés et l'organisation physique des points de stockage ;

- différents coûts : transport, entreposage, immobilisation.

Un deuxième exemple de systèmes logistiques et pas des moindres concerne les systèmes de gestion de la chaîne de soins en santé, appelée encore logistique hospitalière. Nous présentons dans la section suivante les différentes familles de flux qui caractérisent la logistique pour la santé ainsi que les différents processus à contrôler pour assurer une gestion optimisée des patients.

I.4 Logistique hospitalière

I.4.1 L'évolution

Le contexte hospitalier actuel avec la croissance des dépenses de santé incite les établissements de santé à adopter une politique plus rigoureuse et objective de maîtrise des dépenses de santé. L'objectif est double : d'une part le respect des moyens alloués au secteur

de la santé, d'autre part, l'utilisation optimale de ces ressources. Ces nouvelles exigences de productivité sont venues s'ajouter à celles de qualité et de sécurité des soins prodigués aux patients. Ces dernières années, la situation des systèmes de santé rappelle celle du monde industriel et des entreprises de production de biens des années 80. En effet, au cours des vingt dernières années, nous avons assisté à de multiples tentatives d'importation et d'adaptation à l'hôpital de méthodes de gestion empruntées au monde de l'entreprise, et ceci dans le but de profiter de l'expérience gestionnaire dans ce domaine. Les systèmes de santé ont ainsi reconnu de nombreuses restructurations hospitalières. Ces restructurations ont amené, un grand nombre d'hôpitaux à revoir leurs processus opérationnels et à s'investir dans des projets de modernisation de leurs infrastructures et de leurs organisations.

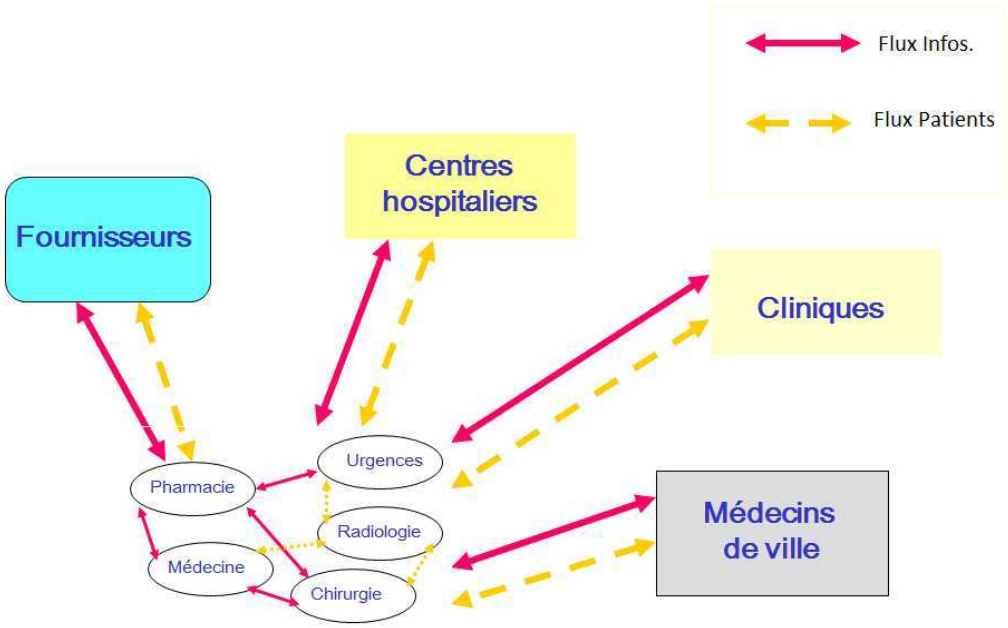

Figure I. 3 Evolution des systèmes de Santé

I.4.2 Les processus de la logistique hospitalière

Le système hospitalier est considéré comme un véritable système de production de soins dont la mission principale est de prodiguer le meilleur soin au patient. Les constats de terrain sur la complexité de l'hôpital sont nombreux et quotidiens : multiplicité des missions, hiérarchisation, cloisonnement, diversité des métiers, hétérogénéité des équipements et des installations (Pham, 2002). On note aussi, un éventail de professions très large. Les nombreuses professions soignantes possèdent en effet une déontologie, une culture professionnelle et une organisation hiérarchique spécifique. Aux professions typiquement hospitalières se rajoute une panoplie d'autres métiers non directement rattachés à l'activité de soins (administratifs, techniciens, informaticiens, etc.).

Traditionnellement, la logistique hospitalière est scindée en deux parties, comme l'illustre le tableau I.1 :

- Les prestations hôtelières qui regroupent les activités à caractère non-médical : repas, blanchisserie, nettoyage, etc.
- La logistique médicotechnique qui regroupe les activités à caractère médical :
    - les prélèvements pour analyse,
    - les transports de produits sanguins,
    - les achats et approvisionnements médicaux (laboratoire, pharmacie, médicaments, dispositifs),
    - des activités annexes comme l'action humanitaire.

Sous-système	Processus métier	Processus logistique
Orienté patient	Consultations, interventions chirurgicales, soins	Transport sanitaire, brancardage, accueil
Dédié pharmacie	Validation et aide aux prescriptions	Acquisition, stockage dispensation et distribution des médicaments, dispositifs médicaux
Dédié linge	Entretien des chambres	Collecte, nettoyage distribution du linge
Dédié restauration	Dispensation et distribution des repas	Acquisition repas ou composants
Soutien administratif	Activités administratives	Acquisitions de fournitures
Soutien technique	Maintenance des équipements	Gestion des connaissances techniques, des produits

Tableau I. 1 Les processus de la logistique hospitalière

I.4.3 Les familles de flux

Le réseau de soins est défini comme étant une collaboration entre les professionnels de santé de différentes disciplines répondant au besoin des patients en matière de santé et d'une prise en charge qualitative. L'objectif du réseau de soin est donc d'améliorer la qualité des soins tout en garantissant leur continuité ainsi que la maîtrise des différents flux et coûts engendrés.
Les flux qui circulent à travers le réseau de soin sont différents et variés.

- les flux de matériels médico-chirurgicaux (par exemple des appareils de radiographie, des seringues, des compresses…) ;
- les flux de médicaments et molécules diverses ;

- les flux d'échantillons et de prélèvements ;
- les flux de matières et matériels hôteliers (plateaux repas, lits, draps, consommables divers...) ;
- les flux de matières et matériels d'entretien et de maintenance ;
- les flux de personnes (patients, personnels et visiteurs).

I.5 Logistique de gestion de crise : Principaux enjeux

En moyenne, chaque année, plus de 210 millions de personnes sont affectées par des catastrophes naturelles. Les organisations humanitaires et militaires doivent donc souvent mettre en place des chaînes logistiques complexes et ce dans un environnement excessivement volatile. Ces chaînes logistiques ont de nombreuses particularités qui les différencient de leurs homologues habituellement rencontrées dans l'industrie traditionnelle. Si l'on exclut certains termes spécifiques à la logistique d'entreprise, tels que client ou magasin, le concept de *SCM* (*Supply Chain Management*) s'accorde avec les objectifs de gestion de l'impact d'une crise. Le concept de *SCM* appliqué aux opérations de secours, ravitaillements et soutien s'appelle la gestion de la crise.

I.5.1 La crise et la gestion de la crise

La crise, qu'elle soit naturelle ou provoquée par l'homme, est une situation à laquelle les gouvernements locaux ne peuvent faire face avec leurs propres ressources. Les crises peuvent être engendrées par plusieurs causes : catastrophes naturelles (telle que les tremblements de terres, les inondations, etc.), conflits politiques ou accidents industriels. Certaines crises vont causer des milliers de morts, d'autres vont toucher un nombre limité de la population. Certaines crises vont durer quelques semaines, comme dans le cas des vagues de chaleur, d'autres peuvent durer des années. Les principaux types de crises sont présentés dans le tableau I.2.

	Type de crise	
Type d'évolution	*Naturelle*	*Provoquée par l'homme*
rapide	Tremblements de terre, tornades, inondations, etc.	Attaques terroristes, coups d'états, accidents nucléaires, etc.
lente	Famine, Sécheresse, Pauvreté, etc.	Conflits politiques, Réfugiés

Tableau I. 2 Les différents types de crises

Chacune de ces situations de crise induit une sortie, un débordement du cadre commun, il est donc indispensable de mettre en place une méthodologie qui permet de revenir rapidement à une situation normale d'avant la crise. C'est ce qu'on appelle la gestion de crise. La gestion de crise et le déploiement d'une chaîne logistique de gestion de crise dépendent de la localisation du désastre, son intensité, sa nature et des acteurs touchés par le sinistre.

La gestion de crise est composée de trois types d'activités :
- La prévention des crises : a pour objectif d'amener le risque de crise à un seuil acceptable et, quand cela est possible, d'éviter que la crise ne se produise effectivement.
- La capacité de réaction opérationnelle : comprend la planification stratégique avancée, les entraînements et la simulation afin d'assurer la disponibilité, la rapidité de mobilisation et de déploiement des ressources nécessaires pour gérer les urgences potentielles.
- La gestion de crise déclarée : il s'agit de la réponse, incluant l'évacuation, la recherche et le sauvetage, au moment de la crise, et le rétablissement de la situation à partir de cette crise, en minimisant ses effets, en limitant les impacts sur l'environnement et la population locale. À plus long terme, l'objectif est de ramener les systèmes à la normale, qu'il s'agisse d'environnement, d'économie ou autres (Jeannin, 2008).

I.5.2 Caractéristiques de la chaîne logistique humanitaire

Une Chaîne Logistique de Gestion de Crise (CLGC), détient le même objectif qu'une chaîne logistique d'une entreprise. Cependant, la logistique humanitaire dans un contexte à forte perturbations est plus difficile à mettre en place qu'une logistique civile. Plusieurs auteurs ont tenté d'identifier les caractéristiques d'un tel environnement. Nous avons retenu cinq grandes catégories :
- Le cycle de vie de l'opération humanitaire :

Le cycle de vie de l'opération humanitaire est composé de 4 phases principales : prédiction et analyse des risques, préparation des plans d'action, intervention immédiate et soutien, et finalement le démantèlement. La durée de chacune de ces phases varie en fonction du type de crise (évolution rapide ou lente).

- L'environnement :

La chaîne logistique de gestion de crise est régie par plusieurs contraintes comme la neutralité, l'impartialité, etc. En d'autres termes, et surtout dans le cas de conflits politiques, tous les acteurs doivent bénéficier du même soutien. La CLGC doit pouvoir assurer la distribution simultanée de ressources vers toutes les zones sinistrées.

- La nature des flux :

Les canaux de distribution gèrent les types de flux traditionnels avec cependant quelques spécificités :

- Les flux physiques sont composés de matières (nourriture, eau, etc.) et d'unités constituées (forces, etc.)
- Les flux d'informations (transmission des ordres, le suivi et la coordination des flux physiques) sont hautement structurés et hiérarchisés.

- Réseau logistique :

L'infrastructure logistique est fixe uniquement en métropole, elle est à créer en opération. Le réseau routier peut souvent faire l'objet de ruptures multiples et non planifiées.

- Les dynamiques :

Les dynamiques d'une CLGC sont très spécifiques puisqu'ils essaient de répondre à certains besoins vitaux. Ces dynamiques sont accentuées dans le cas de crise soudaine, où les besoins surgissent de façon inattendue après la survenance de la crise. Dans la plupart du temps, la CLGC devra :

- Estimer les besoins urgents et vitaux, mais aussi incertains.
- Coordonner l'approvisionnement, urgent et incertain aussi.
- Travailler dans des conditions d'urgence, considérer des évènements imprévisibles, le tout avec des délais très courts.
- Prendre en considération le manque de transparence et le climat instable dans lesquels la CLGC pourrait avoir à fonctionner.

I.5.3 La gestion des flux

Plusieurs types de flux existent, selon le mode de gestion.

I.5.3.1 Flux poussés

Le flux poussé est relatif à une production à partir de besoin estimé, destiné à alimenter un stock ou à le compléter. Dans ce cas de figure, les approvisionnements sont « poussés » au plus loin vers le consommateur. Dans le contexte de gestion de crise, le flux poussé, correspond à la livraison d'une ressource estimée suffisante pour l'opérateur terrain (ou client). En cas de sous-consommation, ces stocks provoqués viendraient handicaper les détenteurs du stock ; en cas de surconsommation, le manque de réactivité serait notoire par manque de visibilité.

I.5.3.2 Flux tirés

Le flux tiré correspond à une production amont pilotée par les besoins de l'aval ; le fournisseur vient compléter, sur demande, le besoin de son client. Dans le contexte de gestion de crise, cela correspond à la demande de réapprovisionnement faite par le consommateur en fonction de ses besoins et en respectant la marge allouée par le commandement. Ce principe prend en compte la prévision des besoins et la notion de stock de sécurité, nécessaire pour pallier une rupture ou une action non judicieuse. Cette méthode assure en théorie l'adéquation permanente du soutien au besoin réel.

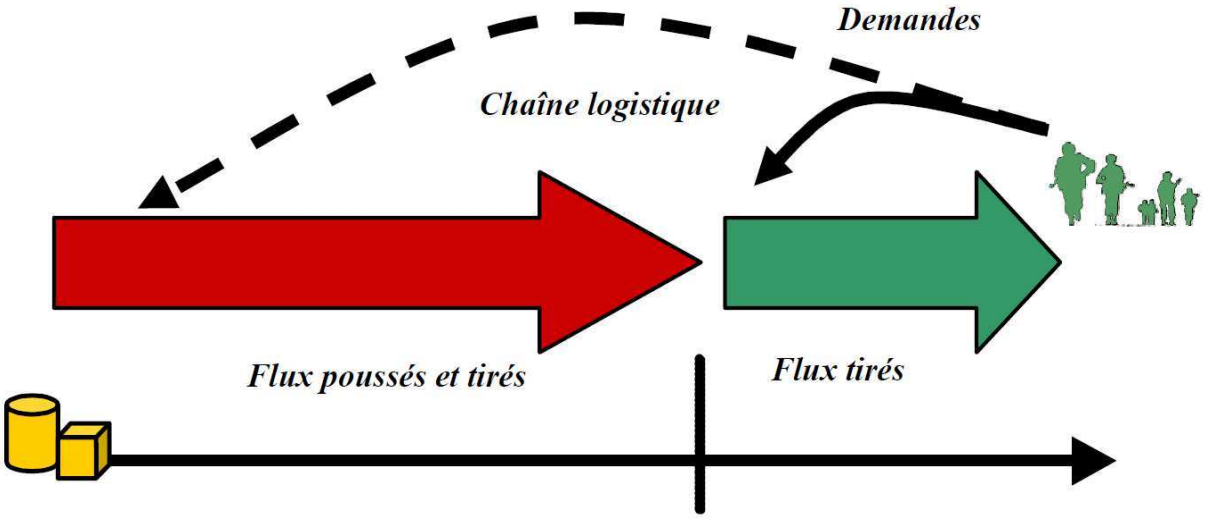

Figure I. 4 Flux poussés et flux tirés

Généralement, en cas de crise, nous utilisons les deux méthodes, comme le montre la figure I.4, avec une préférence pour le flux poussé dans les zones en amont, et le flux tiré dans les zones en aval. En effet, le flux tiré dans les zones en aval a pour but d'adapter réellement les envois à la demande, pour pouvoir coller de près aux exigences du terrain et éviter les ruptures de stock. Mais un fonctionnement à flux tiré en amont présente un risque de retard des ressources, car le délai d'acheminement d'une zone à l'autre est non négligeable. C'est pourquoi les flux poussés sont utilisés en zone amont.

I.5.4 La littérature scientifique pour la gestion de crise

L'évolution du nombre de publications

Le nombre de publications (conférences et revues) comportant « *Humanitarian Supply Chain*» dans leur titre ne cesse d'augmenter avec une accélération depuis 2000.

Le tableau I.3 (Charles, 2010) donne un aperçu sur la quantité de publications sur les chaînes logistiques humanitaires au cours des dernières années. Ceci est une vue partielle de la littérature existante puisque nombreux articles ont été publiés dans des revues liées à la

logistique militaire, aux réfugiés, à la santé mais qui n'apparaissent pas toujours dans les bases de données universitaires.

Mots clés	*Les bases de données universitaires*			
	Science Direct *Recherche dans tout le contenu*	Science Direct *Recherche dans le titre, le résumé et les mots clés*	ISI Web *Recherche dans le thème*	Springerlink *Recherche dans tout le contenu*
Humanitarian Supply Chain	665	9	15	447
Humanitarian Logistics	625	17	34	440
Humanitarian Supply Chain and Logistics	147	7	10	115
Total articles Humanitarian Supply Chain and/or Logistics	1143	19	39	772

Tableau I. 3 Résumé de la littérature existante sur « humanitarian supply chains/logistics »

La figure I.5 présente la répartition de la littérature publiée selon les différentes étapes du cycle de vie de la crise. Environ 44% de la recherche publiée concerne l'atténuation et l'analyse des risques. L'intervention immédiate et soutien et la préparation viennent respectivement en 2ème et 3ème position avec 23,9% et 21.1% des articles publiés. Mais le domaine le plus en besoin de recherche est la phase de démantèlement ou rétablissement après le sinistre avec seulement 11% des articles publiés.

La figure I.5 présente une répartition des articles en fonction de leurs contributions depuis l'année 2000 : dans le domaine de la modélisation, du développement de la théorie et du développement d'applications et d'outils. Nous remarquons alors que plus de la moitié de la recherche publiée dans le domaine de la gestion de crise concerne la modélisation, suivie par 26.6% sur le développement de la théorie et 15.6% concernant le développement d'outils et applications.

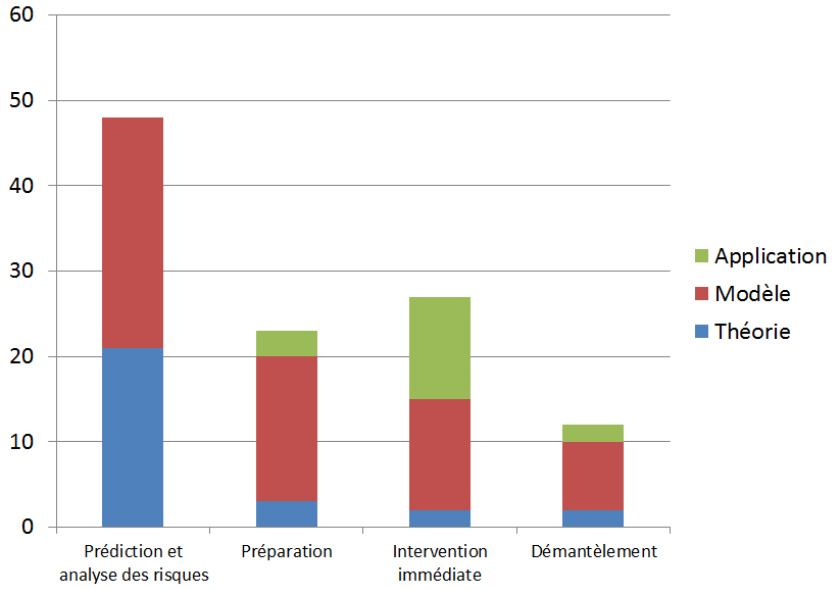

Figure I. 5 Répartition des publications selon leurs contribution (N. Altay et W. G. Green)

D'après les analyses rapportées dans cette section, un nombre croissant d'articles ont été publiés sur le thème de la gestion du désastre. La plupart des travaux actuels sur les chaînes logistiques de gestion de crise s'accordent sur le fait que les recherches futures devraient porter sur des méthodes pour l'amélioration de la gestion des stocks, la conception des systèmes de distribution, la coopération et la coordination et la mesure de la performance (meilleurs échanges d'informations, recherche de la meilleure quantité économique de transport, négociation entre acteurs,…).

En effet, un constat de plus en plus évident est le fait que la chaîne logistique de gestion de crise n'est pas seulement une relation dyadique ni même un ensemble de relations limitées à un ou deux rangs. Une CLGC est bien souvent un réseau d'entités géographiquement distribuées plus vaste et donc plus complexe à gérer. Nous comparons dans la section suivante les différentes façons de piloter la chaîne logistique, à savoir pilotage centralisé versus pilotage distribué, afin d'améliorer le processus décisionnel dans les chaînes logistiques complexes de type chaîne logistique de gestion de crise.

I.6 Besoins méthodologiques pour les chaînes logistiques

Dans cette section, nous présentons rapidement trois grands axes de recherche en termes de méthodologies et techniques utilisées au service de la gestion des chaînes logistiques, à savoir la modélisation, l'optimisation et l'aide à la décision.

1.6.1 Modélisation

Il existe différents modèles de chaînes logistiques. Le choix du modèle dépend directement du type de problème et de structure que l'on veut étudier.

I.6.1.1 Modèle analytique

Les modèles analytiques permettent de décrire le système par un ensemble d'équations mathématiques. Pour qu'un modèle analytique (déterministe ou stochastique) soit viable, il faut qu'il soit relativement simple, c'est-à-dire qu'il faut faire un certain nombre d'hypothèses et de simplifications. C'est peut-être pour cela que les modèles analytiques se contentent généralement d'aspects basés sur la distribution dans une structure dyadique (Huang et al., 2003).

I.6.1.2 Modèle par simulation

Le modèle par simulation est généralement utilisé lorsqu'il n'existe pas une relation entre les différentes variables du système et ne pouvant donc pas se mettre sous la forme d'un modèle analytique. Les modèles de simulation sont des modèles à la fois stochastiques et dynamiques. Les travaux d'analyse des processus décisionnels par simulation de modèle sont nombreux. (Maria, 1997) propose de classer les différentes techniques de simulation de chaînes logistiques en quatre grandes parties : i) la simulation par tableur, ii) l'approche par la dynamique des systèmes, iii) l'approche par événements discrets ou simplement iv) les jeux d'entreprises, qui permettent d'éduquer et d'entraîner les utilisateurs à certains aspects de la gestion de la chaîne logistique.

1.6.2 Optimisation

L'optimalité de la qualité de service en logistique sous-entend une satisfaction des utilisateurs du système proposé de par l'intégration des fonctionnalités nécessaires à la bonne gestion de la chaîne logistique. Dans ce sens, le concept d'optimisation a retenu toute notre attention.

L'optimisation a été introduite dans un souci d'amélioration des services fournis peu importe le domaine auquel ils s'appliquent. Un problème d'optimisation concerne l'exécution de méthodes spécifiques en quête d'un optimum. Ce dernier peut être une valeur maximisant ou minimisant une fonction *f*, dite fonction objectif ou fonction de coût ; elle est encore appelée critère d'optimisation (Sghaier, 2011). Selon le cas, et qu'il s'agisse d'un problème mono variable ou multi variable, continu ou discret, etc., une méthode d'optimisation adéquate est choisie pour résoudre le problème posé. Entre méthodes exactes, méta-heuristiques, hybrides ou autres, chercheurs et praticiens ont à leurs dispositions un large panel de choix de

méthodes d'optimisation qu'ils peuvent adopter. Toutes ne sont pas appropriées au problème d'optimisation auquel ils doivent faire face et un choix doit donc être fait. A défaut de quoi, l'optimisation et l'efficacité escomptées s'en trouveront entravées.

1.6.3 Aide à la décision

Les techniques d'aide à la décision ont pour but de modéliser de manière la plus fidèle possible les préférences d'un expert. Cette modélisation va permettre alors de concevoir et de construire des outils adaptés et capables d'assister ou de remplacer un décideur sur des problématiques complexes. L'aide à la décision formalise l'expertise obtenue après un entretien avec le décideur et les interactions entre le décideur et son environnement.

Étant donné que nous nous intéressons principalement à l'expertise des décideurs dans le domaine de la logistique, le modèle utilisé se doit d'être souple et suffisamment élaboré afin de permettre la représentation des différents comportements décisionnels les plus communément rencontrés en gestion de chaînes logistiques de gestion de crise.

I.7 Pilotage centralisé vs pilotage distribué

Nous consacrons la première partie de cette section à la présentation de l'évolution des systèmes centralisés. Ces derniers, initialement de type MRP2 (Cf. I.7.2.1), ont évolué et ont donné naissance aux systèmes APS *(Advanced Planning and Scheduling*). Nous verrons que ces systèmes sont principalement basés sur la centralisation de la prise de décision et que leur développement est principalement marqué par l'augmentation croissante du nombre de fonctions comprises. Notons que les travaux de recherche associés à ce type de systèmes centralisés sont toujours aussi nombreux jusqu'à notre époque, et que ceux-ci correspondent aux systèmes plus amplement mis en application dans l'industrie. Les entreprises avaient installé ce type de systèmes essentiellement pour des raisons de coût, mais également à cause de leur incapacité à obtenir des informations détaillées, en temps réel, issues des différentes entités composant leur chaîne logistique.

Ensuite, nous présentons le courant de recherche qui s'intéresse aux systèmes intelligents de distribution ou de logistique. La vision de la prise de décision logistique au niveau de ces nouveaux systèmes (au début de 1990) a changé radicalement en passant de la perspective centralisée conventionnelle à une perspective distribuée. Ce changement de perspective est basé sur les nouvelles nécessités en termes de flexibilité et de réactivité. Ainsi, ces nouveaux systèmes se proposent de donner une réponse à des problématiques comme : la distributivité, la re-configurabilité, l'interopérabilité, la réutilisabilité, etc.

I.7.1 Définition du pilotage

Le pilotage d'une chaîne logistique composée de plusieurs entités est une fonction qui permet d'exploiter les ressources qui sont à disposition dans cette chaîne logistique. Cette fonction trouve sa matérialisation dans deux éléments d'organisation : le système d'information et le système décisionnel. Le système d'information permet de stocker, adapter et mettre à disposition les données sur lesquelles se basent les centres de décision. Le rôle du système d'information est de fournir à chaque centre de décision l'information nécessaire et suffisante à une prise de décision adéquate.

I.7.2 Les systèmes centralisés conventionnels

I.7.2.1 Les systèmes MRP2

Le MRP2 (*Manufacturing Ressource Planning*) est défini par l'APICS (Association Internationale pour le Management des Opérations) comme une méthode de planification efficace de toutes les ressources d'une entreprise industrielle. Les systèmes MRP2 offrent plusieurs services dont les plus utilisés sont les plannings opérationnels, les plans financiers et surtout l'aide à la décision par l'analyse de scénarios. Cette méthode est constituée d'un grand nombre de fonctions toutes liées entre elles : Plan Stratégique (PS), Plan Industriel et Commercial (PIC), Programme Directeur de Production (PDP), calcul des besoins, planification des besoins en capacité et suivi de l'exécution des plans.

I.7.2.2 Les ERP

Basés sur les fondations technologiques du MRP2, les progiciels intégrés ERP (*Entreprise Resource Planning*) permettent de gérer les fonctions transactionnelles de l'entreprise. Les systèmes ERP couvrent plusieurs fonctions comme la fabrication, la distribution, la comptabilité, la gestion des ressources humaines, les finances, la gestion de projets, la gestion de stocks, la maintenance et le transport. Herrera (Herrera, 2011) présente les avantages communément promus par les entreprises qui développent des systèmes ERP:

a) Permettre l'accès à des informations fiables (base de données commune, consolidation des données et amélioration des rapports).
b) Eviter la redondance d'opérations et de données (les modules accèdent aux mêmes données dans la base centrale).
c) Eviter les multiples opérations de mise à jour et d'entrée des données.
d) Proposer des fonctions de commerce électronique (commerce par Internet et culture de collaboration).

e) Proposer des modules additionnels tels que la gestion de la relation client et le SCM.

D'autres avantages peuvent être dégagés comme : la réduction des temps de cycle et de livraison, la minimisation des coûts et l'adaptabilité, mais ils sont encore discutables à ce jour. Les inconvénients les plus constatés de ces systèmes sont : le temps dépensé pour l'implémentation, le coût du système, mais surtout la rigidité à l'égard de la complexité des différents processus de l'entreprise.

I.7.2.3 Les systèmes de planification avancée

L'idée inhérente aux APS (*Advanced Planning and Scheduling* ou *Advanced Planning Systems*) est l'utilisation des techniques et des algorithmes qui permettent d'obtenir des solutions optimisées à différents niveaux de décision (Herrera, 2011).

Les progiciels APS assurent la planification et l'optimisation de la circulation des flux le long d'une chaîne logistique. Il s'agit de techniques qui rentrent dans le cadre de l'analyse et de la planification de la production et de la logistique à court, moyen et long termes. Ils sont présentés comme la version évoluée de l'ERP vu qu'ils fournissent une solution qui optimise le travail collaboratif en fonction d'objectifs tels que le taux de service et les marges sur les activités. Les composantes principales d'un système APS sont la planification de la production, l'ordonnancement de la production et la planification de la distribution et du transport. Quelques inconvénients sont à citer, notamment ceux liés à la flexibilité, la re-configurabilité, la robustesse, l'agilité, la réactivité ou la réutilisation. En effet, ces problèmes sont encore loin d'être résolus par ce genre de systèmes en raison de la complexité engendrée.

I.7.2.4 Quelques Critiques

La nature combinatoire de la plupart des problèmes dans les systèmes de distribution, tels que le problème d'allocation des ressources, conduit à la simplification de la complexité inhérente à ce genre de systèmes. En effet, l'information nécessaire pour l'obtention des décisions aux différents niveaux est une information agrégée et dans ce processus d'agrégation, d'importantes relations entre les différents composants du système peuvent être négligées.

Un deuxième point faible des approches centralisées est l'absence d'un vrai retour d'information depuis les niveaux inférieurs de la chaîne logistique, ce qui réduit la réactivité et la flexibilité vis-à vis des perturbations possibles.

I.7.3 Les systèmes distribués intelligents

I.7.3.1 Les systèmes fractals

Les systèmes fractals ont été proposés à partir des travaux de (Warnecke, 1993). Ces systèmes reprennent les caractéristiques de l'auto-similarité et de la récursivité utilisées dans la géométrie fractale, pour la conception d'une structure d'un système. Ce système est construit en assurant la cohérence entre les objectifs globaux et locaux des entités qui le composent. L'objectif est de doter la structure du système de la réactivité et la souplesse nécessaire pour résister aux changements de l'environnement.

I.7.3.2 Les systèmes multi agents

Au-delà des modélisations prenant en compte, dans un modèle analytique, l'ensemble de la chaîne logistique, des approches consistent à mettre en avant l'aspect distribué de la chaîne en créant des maillons possédant plus ou moins d'autonomie. Ces modélisations relèvent des concepts orientés objets et se traduisent par une architecture distribuée à base d'acteurs (agents). Le paradigme des systèmes multi-agents apparaît comme une réponse au problème de la modélisation de systèmes distribués à grande échelle. Les systèmes multi-agents seront présentés dans le chapitre II ainsi que leurs applications dans le domaine de la logistique.

I.7.3.3 Les systèmes holoniques

À partir des concepts exprimés par Alfred Koestler dans son livre (Koestler, 1968), le HMS (*Holonic Manufacturing Systems*) consortium a proposé les systèmes holoniques de production. Une entité holonique possède une structure récursive de sociétés. Les holons possèdent des propriétés comme l'autonomie et la coopération et la conservation d'une flexibilité dynamique au niveau distribué. Chaque holon du système possède les mêmes caractéristiques que l'ensemble et possède une spécialité. Depuis leurs utilisations dans les systèmes manufacturiers, les systèmes holoniques ont été appliqués dans de nombreux autres domaines tels que la santé (Ulieru & Geras, 2002), les systèmes de transport (Bürckert et al., 1998) et les robots footballeurs (Barbat et al., 2001).

I.7.3.4 Les systèmes contrôlés par le produit

Un système contrôlé par le produit fait évoluer la vision précédente vers un système plus interopérable et intelligent. En effet, le produit devient le contrôleur des ressources de l'entreprise. C'est ce qu'on appelle un produit intelligent. Les systèmes contrôlés par le produit ont été généralement perçus comme une classe particulière des systèmes holoniques. L'idée fondamentale est le changement de perspective depuis une vision classique

hiérarchique et agrégée de la fonction de planification et de pilotage de la production vers une perspective distribuée de la prise de décision. Dans ce sens, une partie de la décision est effectuée en local et tout au long du cycle de vie des produits.

I.7.4 Motivations pour la décentralisation du pilotage dans le cas de CLGC

Herrera (Herrera, 2011) propose plusieurs motivations pour la décentralisation du pilotage :

Faisabilité

Dans certains cas, la prise de décision distribuée est la seule solution automatisée réalisable. Par exemple, dans les chaînes logistiques où les entités sont en concurrence les unes avec les autres et où les informations partagées sont rares pour des raisons de sécurité, la décentralisation est une stratégie plus systématique et automatisée pour remplacer les pratiques ad-hoc existantes.

Robustesse et flexibilité

Cet avantage permet d'améliorer la performance pour différents scénarios. C'est l'une des caractéristiques les plus importantes pour justifier l'utilisation des systèmes à base d'agents. Spécifiquement, cela signifie que le système a comme propriétés : la robustesse aux pannes ou aux perturbations des plans, la réorganisation en ligne et la réponse effective aux perturbations externes.

Reconfigurabilité

Les solutions basées sur des agents soutiennent fortement une approche *plug and play*. Cela permet de changer, ajouter ou enlever, autant le matériel que les modules de logiciel, en ligne à chaque fois que cela est nécessaire. Cela rend aussi la maintenance du système significativement moins chère. Ce changement est la conséquence des échecs d'équipement ou la conséquence d'un changement de plan. La migration depuis l'ancienne technologie peut évoluer lentement sans arrêter l'opération

I.8 Terrain d'étude – partenaire industriel

Leader mondial des secteurs de l'aéronautique, de l'espace, de la défense, de la sécurité et des services associés, le groupe EADS (*European Aeronautic Defence and Space company*) comprend l'avionneur Airbus, le plus grand hélicoptériste mondial Eurocopter et la joint-venture MBDA, leader mondial dans le domaine des missiles. EADS est également le partenaire majeur du consortium Eurofighter, le maître d'œuvre du lanceur d'Ariane2, l'architecte de l'avion de transport militaire A400M et le plus grand partenaire industriel pour le système européen de navigation par satellite Galileo.

En 2011, le Groupe – qui comprend Airbus, Astrium, Cassidian et Eurocopter – a dégagé un chiffre d'affaires de 49,1 milliards d'euros, avec un effectif de plus de 133 000 personnes , comme l'illustre la figure I.6.

Depuis sa création en l'an 2000, le groupe a toujours été à l'avant-garde de l'innovation et de l'internationalisation au sein de l'industrie aérospatiale et de défense. Constamment à l'écoute des besoins émergents de ses clients à la fois civils et militaires, EADS améliore sans cesse ses modes de fonctionnement.

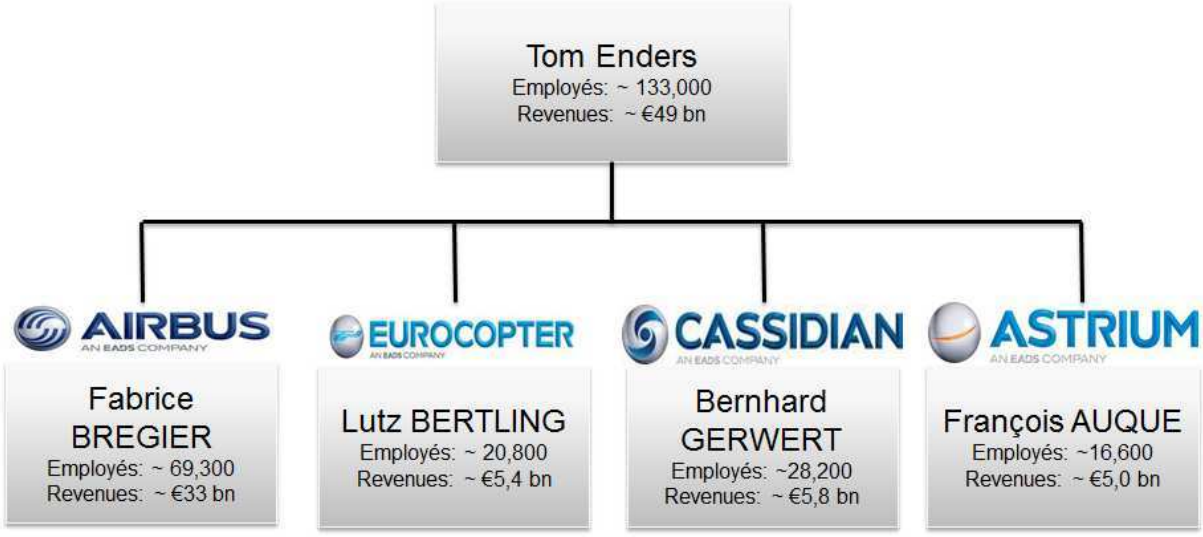

Figure I. 6 Organigramme d'EADS (en 2011)

Cassidian, division du groupe EADS, est un leader mondial dans le domaine des solutions et systèmes de sécurité intégrés et de l'intégration de grands systèmes, proposant à ses clients civils et militaires dans le monde entier des produits et services à forte valeur ajoutée : systèmes aériens (aéronefs et systèmes de drones), systèmes navals, terrestres et « interarmées », renseignement et surveillance, cyber sécurité, communication sécurisée, systèmes de test, missiles, services et support. Avec quelque 28 000 employés, Cassidian a réalisé en 2011 un chiffre d'affaires de 5,8 milliards d'euros.

Véritable pôle Défense et Sécurité d'EADS, Cassidian représente l'entité chef de file du groupe pour le développement de solutions de systèmes. Cette division a bâti sa réputation sur une longue et puissante lignée de systèmes d'armes et de missiles aéroportés, et a incorporé les capacités réseau-centrées de toute dernière génération en s'appuyant sur trois facteurs clés: la veille technologique, l'intégration et l'expertise. En misant sur l'aspect critique et sécuritaire des missions, Cassidian prépare ses clients à relever leurs nouveaux défis, que ce soit sur terre, en mer ou dans les airs.

La thèse est réalisée en coopération avec le département « ***Logistics Information Systems*** » qui fait partie de la « *business unit Defence and Communication Systems* » (DCS), de la division Cassidian d'EADS.

I.9 Positionnement de la thèse

L'outil le plus utilisé pour la validation des systèmes décentralisés tels les chaînes logistiques de gestion de crise est la simulation. La méthode la plus généralement adoptée consiste à construire un modèle du système et ensuite de comparer le résultat généré par celui-ci avec la situation réelle et à l'évaluer par rapport à différents critères de performances. La proposition de recherche que nous faisons consiste plus précisément en un ensemble d'outils et de démarches permettant l'optimisation des flux dans une CLGC. Nous avons opté pour une modélisation de la CLGC à base d'agents communicants. Dans ce modèle, les agents qui représentent les différents acteurs de la chaîne logistique, sont en liaison directe avec ce qui se passe sur le terrain, ils pompent les informations en continu de la couche terrain et comparent la situation réelle à la situation logistique de référence. En fonction de ces informations ainsi que des divers modèles mathématiques à disposition, ces agents vont devoir adapter leurs comportements afin de réagir au mieux aux différentes perturbations du terrain, dans le but de revenir à une situation d'avant-crise.

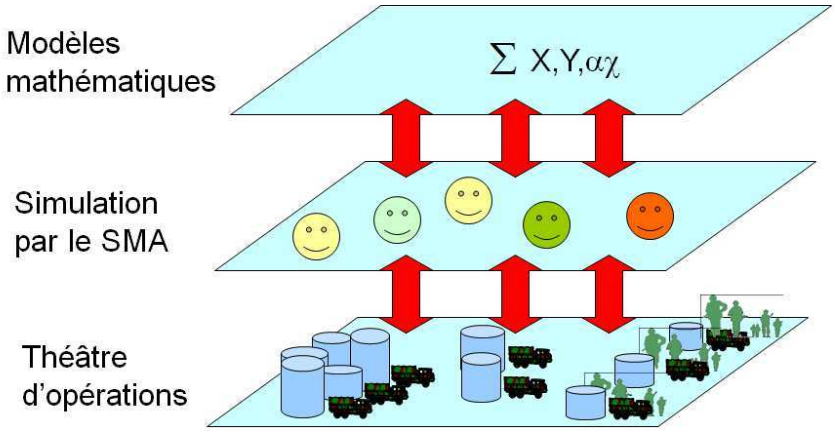

Figure I. 7 Architecture à trois couche de notre système

I.10 Conclusion

Nous avons présenté dans ce chapitre les éléments de base de la chaîne logistique et de sa gestion afin de bien assimiler ce vaste domaine. Nous avons mis en évidence quelques travaux rencontrés dans la littérature et qui peuvent être considérés comme étant le contexte général de notre travail de thèse. Une des motivations majeures de ce travail de recherche est d'optimiser les flux logistiques dans une chaîne de gestion de crise à base d'agents communicants. Le problème étudié est donc à cheval entre ces deux classes de problèmes :

d’une part, il s’intègre dans une approche multi-agent pour la modélisation du fonctionnement de la chaîne logistique, d’autre part, il s’apparente à un problème global d’optimisation.

Nous proposons dans le chapitre suivant un état de l’art des méthodes d’optimisation et du paradigme multi-agent, avant de démontrer toute l’originalité de leur alliance.

Chapitre II : L'Alliance entre les Systèmes multi-agents et l'Optimisation

II.1 Introduction

Sur la base des problématiques précédemment énoncées, (cf. Chapitre I), nous orientons notre étude vers les fondements essentiels de l'optimisation. Dans notre cas, des notions très importantes et d'actualité telles que les systèmes multi-agents (SMA) et l'optimisation distribuée sont directement impliquées dans notre approche. Leur combinaison efficace pour la réduction de la complexité du problème de gestion de crise nous mène à y porter l'intérêt qu'il se doit. Nous commençons par définir quelques concepts de base de l'optimisation, puis nous présentons un état de l'art sur les différentes méthodes d'optimisation. Ensuite, notre intérêt se porte sur les systèmes multi-agents, qui constituent un concept pour la mise en place d'une méthodologie de résolution et d'optimisation distribuée.

II.2 Optimisation

Dans le domaine de gestion des chaînes logistiques, nous cherchons toujours à maximiser les bénéfices et minimiser les pénalités dues aux retards. Ces objectifs à atteindre s'inscrivent dans le cadre des problèmes d'optimisation. Plusieurs méthodes sont développées afin d'aider à atteindre l'optimalité relativement aux décisions que les logisticiens doivent prendre (méthodes exactes, heuristiques et méta heuristiques).

II.2.1 Quelques concepts

II.2.1.1 Définition d'un problème d'optimisation

Un problème d'optimisation concerne l'exécution de méthodes spécifiques en quête d'un optimum. Ce dernier peut être une valeur maximisant ou minimisant une fonction *f*, dite fonction objectif ou fonction de coût ; elle est encore appelée critère d'optimisation (Sghaier, 2011). Il s'agit donc de trouver parmi l'ensemble des variables de décision, l'élément qui optimise ce critère.

M. Ejday (Ejday, 2011) définit le concept d'optimisation comme comprenant deux phases principales : la modélisation et la résolution du problème.

La phase de modélisation est elle-même divisée en trois étapes :

- L'identification des paramètres sur lesquels l'utilisateur va agir pour faire évoluer le système, appelés variables de décisions, que nous pouvons désigner par un vecteur $\vec{u} \in \Re^n$.

- La définition de la fonction coût ou fonction objectif qui va permettre d'évaluer l'état du système.

- La modélisation des contraintes auxquelles sont soumises les variables de décision.

La deuxième phase de résolution consiste à rechercher la valeur optimisant la fonction coût en appliquant l'algorithme d'optimisation choisi à cet effet. Le problème d'optimisation consiste alors à déterminer les variables de décision conduisant aux meilleures conditions de fonctionnement du système.

Un problème d'optimisation peut s'écrire de la façon suivante :

$$minimiser\ f(\vec{u}), \qquad sous\ la\ contrainte\ \vec{u} \in U \ \ o\grave{u}\ U\ est\ un\ sous\ ensemble\ de\ \Re^n$$

L'ensemble *U* est appelé *ensemble des contraintes*. Il est souvent défini par des égalités et des inégalités : $U = \{ \vec{u} \in \Re^n : h(\vec{u}) = 0, g(\vec{u}) \leq 0\}$ *avec* $g \in \Re^m$ et $h \in \Re^p$.

Nous distinguons trois types différents d'optimums (minimums ou maximums) : un *Minimum global*, un *Minimum local fort* et un *Minimum local faible* $\vec{u}^*$:

- Le minimum global : Trouver le minimum global revient à trouver une solution dont l'optimalité s'étend sur tout l'espace de recherche considéré.

$$f(\vec{u}^*) < f(\vec{u}) \quad \forall \vec{u}^* \neq \vec{u}$$

- Le minimum local fort : Le point $\vec{u}^*$ s'appelle le *minimum local fort* de la fonction *f* quand il existe un voisinage $V(\vec{u}^*)$ tel que $\vec{u} \in V(\vec{u}^*)$.

$$f(\vec{u}^*) < f(\vec{u}) \ \forall \vec{u}^* \neq \vec{u}$$

- Le minimum local faible : Il reprend un peu la définition d'un minimum local fort avec cependant l'existence de points de l'espace ayant la même valeur de *f* que lui. Le point $\vec{u}^*$ s'appelle le *minimum local faible* de la fonction *f* quand il existe un voisinage $V(\vec{u}^*)$ tel que $\vec{u} \in V(\vec{u}^*)$ tel que

$$f(\vec{u}^*) \leq f(\vec{u}) \ \forall \vec{u}^* \neq \vec{u}$$

Indépendamment de l'approche, une bonne méthode d'optimisation doit considérer les points suivants :

• la robustesse : la méthode doit être applicable pour une large classe de problèmes, indépendamment des paramètres et des valeurs initiales,

• l’efficacité : elle ne doit pas être relativement chère en temps CPU et en mémoire vive,

• la précision : le résultat numérique doit bien approcher la vraie solution du problème de minimisation.

II.2.1.2 La notion d’algorithme

Un algorithme est défini comme une suite d'instructions élémentaires destinées à résoudre un problème (Toussaint, 2010).

La performance d’un algorithme appliqué à un jeu de donné est mesurée selon différents critères :

- le temps d'exécution : il représente le temps nécessaire à l’algorithme pour calculer la solution;

- l'espace mémoire : il représente la quantité de mémoire nécessaire à l’algorithme pour calculer la solution à partir du jeu de données de départ ;

- l'erreur : elle représente l'erreur éventuelle qui peut exister entre la solution calculée et la solution optimale. Notons qu'une telle erreur ne peut être mesurée que si l'on possède une métrique sur l'espace des solutions ;

- la robustesse : elle exprime la capacité de l’algorithme à s'adapter à des variations sur les jeux de données (de manière intuitive, un algorithme est robuste si un faible changement dans les données n'induit pas ou peu de changements dans la solution. Au contraire, si la solution doit être entièrement recalculée alors l'algorithme n'est pas robuste).

Pour les études de complexité, il est nécessaire de définir la taille d'une instance. De manière générale, nous dirons qu'une instance est de taille n si l'instance occupe n cases mémoires. La complexité d'un algorithme est alors une fonction de n (Toussaint, 2010). On définit la complexité C_A d'un algorithme A comme le nombre maximal d'instructions élémentaires effectuées pour traiter n'importe quelle instance, de taille n fixée, d'un problème. Il en découle les notions de « polynomialité » et d' « exponentialité » : A est dit polynomial, s'il existe un polynôme P tel que : $\forall\, n \in \mathbb{N}$, $C_A(n) \leq P(n)$. Dans le cas contraire A est dit exponentiel. La complexité d'un algorithme s'exprime alors souvent par un ordre de grandeur dépendant de la taille n des données :

• $O(1)$: complexité constante (indépendante de la taille de la donnée)
• $O(log(n))$: complexité logarithmique
• $O(n)$: complexité linéaire

• $O(n^p)$: complexité polynomiale ($p \in \mathbb{N}$, $p \geq 2$)

• $O(a^n)$: complexité exponentielle ($a \in \mathbb{R}$, $a > 1$)

II.2.1.3 Classification des problèmes d'optimisation

La complexité d'un problème est donnée par rapport à la plus faible complexité de l'algorithme qui résout ce problème. Notons que cette approche est délicate puisqu'il peut exister plusieurs algorithmes, éventuellement de complexité différente, pour résoudre un même problème. Nous donnons ici un aperçu général des grandes classes de complexité qui concernent les problèmes de décision.

• La classe *P*

Un problème *P* est dit polynomial (ou polynomial-temps), s'il existe un algorithme de complexité polynomiale qui résout *P*. L'ensemble des problèmes polynomiaux forme la classe *P*.

• La classe *P-Espace*

Un problème *P* est dit polynomial-espace, s'il existe un algorithme qui résout *P* en espace polynomial par rapport à la taille de ses données.

• La classe *NP*

Un problème *P* est dans *NP* si pour toute instance de ce problème on peut vérifier que l'instance est solution du problème en un temps polynomial. Cette catégorie est fondamentale car elle contient la plupart des problèmes d'optimisation combinatoire. Remarquons que, pour n'importe quelle instance d'un problème polynomial, il est toujours possible de vérifier en temps polynomial si elle est solution de ce problème, donc $P \subseteq NP$.

• Les problèmes *NP-complet*

Une notion essentielle pour définir ce qu'est un problème NP-complet est celui de la dominance : Un problème *P* domine un problème *P'* (ou encore *P'* est réductible polynomialement en *P*) si :

1. On peut transformer toute instance *I* de *P* en une instance *I'* de *P'* grâce à un algorithme polynomial ;
2. *I* est solution de *P* si et seulement si *I'* est solution de *P'*.

Il découle de cette définition qu'un problème *NP-complet* est un problème de la classe *NP* qui domine tous les autres. La notion de NP-complétude est donc associée à une notion de dominance et non, comme il est parfois dit abusivement, à une notion d' « exponentialité »

(Toussaint, 2010). Afin de résoudre ces problèmes, on utilise différents types d'algorithmes. Nous pouvons en citer:

- Les algorithmes approchés (appelés heuristiques), qui permettent de trouver des solutions approchées dans des temps de calcul raisonnables, mais sans avoir d'indication sur la qualité de la solution trouvée ;
- Les algorithmes d'approximation avec garantie, qui permettent de trouver des solutions approchées et garantissent une qualité de la solution fournie (sous forme d'écart maximal par rapport à la solution optimale) pour toutes les instances du problème ;
- Les algorithmes exacts, comme l'exploration arborescente ou la programmation linéaire qui, couplés avec des mécanismes de filtrage, peuvent s'avérer très performants.

Ces méthodes fournissent des solutions exactes mais le temps de calcul n'est pas borné polynomialement. C'est pourquoi, elles sont souvent réservées à des instances de taille modérée.

Dans la seconde section de ce chapitre, nous donnons un aperçu des heuristiques et métaheuristiques classiques, puis dans la troisième section nous nous intéressons aux algorithmes exacts.

II.2.2 Heuristiques et métaheuristiques

Cette première approche consiste à utiliser des méthodes heuristiques visant à identifier rapidement de bonnes solutions. Une heuristique permet alors d'identifier au moins une solution réalisable pour un problème d'optimisation, mais sans garantir que cette solution soit optimale.

Il n'existe pas, à notre connaissance, de définition officielle. Cependant il est communément admis que :

- Une heuristique désigne un algorithme qui résout un problème d'optimisation donné, sans garantie d'optimalité mais dans des temps de calcul raisonnables (un exemple connu est l'algorithme glouton). Les méthodes heuristiques sont souvent classées en deux catégories : les méthodes constructives permettant de construire une solution réalisable et les méthodes d'amélioration permettant de visiter plusieurs solutions réalisables en tentant d'améliorer la valeur de l'objectif ;

- Une métaheuristique désigne un schéma algorithmique général qui peut s'appliquer à différents problèmes d'optimisation combinatoire. Plus précisément, elle utilise des stratégies qui guident la recherche dans l'espace des solutions, ces stratégies étant indépendantes du

problème auquel on les applique. Le but est d'explorer le plus efficacement possible l'espace des solutions afin de ne pas rester bloqué dans les minima locaux et de se diriger rapidement vers les régions les plus prometteuses. Il existe un grand nombre de métaheuristiques allant de schémas très simples (qui mettent en œuvre des processus de recherche basiques, comme la descente), à des schémas beaucoup plus complexes (avec des processus de recherche élaborés comme les colonies de fourmis).

II.2.2.1 Les algorithmes gloutons

Les algorithmes gloutons (*greedy algorithms*) sont parmi les schémas heuristiques les plus simples et les plus rapides. Ils construisent une solution de manière itérative sans jamais remettre en cause les décisions prises dans l'itération antérieure. Dans les cas où l'algorithme ne fournit pas systématiquement la solution optimale, il est appelé une heuristique gloutonne.

Algorithme glouton
In: S (séquence)
Out: A
1: **début**
2: S= Ø
3: **Tant que** (S incomplète) faire
4: Construire la liste *L* des éléments insérables dans *S* ;
5: **Si** $L \neq \emptyset$ **alors**
6: Evaluer le coût incrémental des éléments de *L* ;
7: Insérer l'élément avec le coût minimum ;
8: **Return** *S*
9: **fin**

Les métaheuristiques que nous présentons dans les sections suivantes proposent des stratégies de recherche plus élaborées. Certaines de ces stratégies ont été conçues par analogie avec d'autres domaines, comme par exemple la physique (recuit simulé) ou la biologie (algorithmes évolutionnaires, colonies de fourmis). La plupart des métaheuristiques présentent l'inconvénient d'avoir de nombreux paramètres à régler dont dépend le compromis délicat entre efficacité et temps de calcul. Nous présentons les métaheuristiques les plus connues, à partir desquelles de nombreuses variantes peuvent être élaborées.

II.2.2.2 Le recuit simulé (Simulated Annealing - SA)

Cette métaheuristique est caractérisée par la présence d'une variable de contrôle appelée température (par analogie aux processus thermodynamiques dont elle s'inspire) qui fixe les conditions dans lesquelles une transformation dégradante est acceptée. Siarry (Siarry, 2002) explique le fonctionnement général de cette méthode qui s'appuie sur l'algorithme de

Metropolis (Metropolis et al., 1953). Disposant d'une configuration initiale, une certaine température *T* suit le procédé qui repose sur le fait de faire subir au système des modifications élémentaires. A chaque itération, si la modification en question arbore une amélioration qui converge vers l'objectif fixé (i.e. diminution de la fonction objectif (ou énergie)), la modification est acceptée. Par ailleurs, même si cette même modification provoque au contraire une détérioration ΔE, il est envisageable qu'elle soit acceptée mais avec une certaine probabilité de $\exp(-\Delta E/T)$. Cette démarche est ensuite renouvelée en gardant la température constante, jusqu'à ce que l'équilibre thermodynamique soit atteint. Toute la démarche ainsi décrite est alors renouvelée pour une nouvelle valeur de la température plus basse et une série de transformations est alors effectuée. La méthode du recuit simulé est souple vis-à-vis des évolutions du problème et facile à implémenter, elle est décrite génériquement dans la figure II.1.

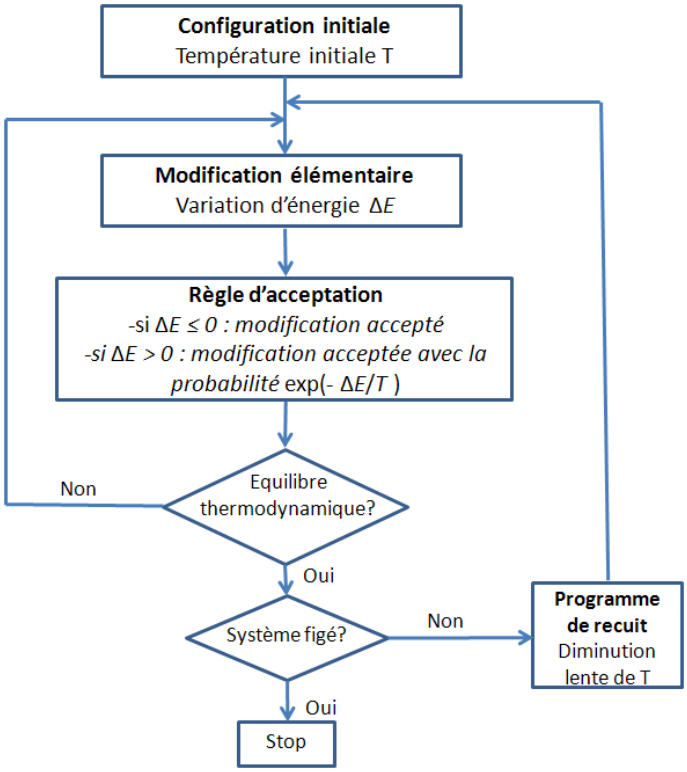

Figure II. 1 Organigramme de la métaheuristique du Recuit Simulé (Siarry, 2002)

II.2.2.3 La recherche tabou (Tabu Search – TS)

La recherche tabou est également une des métaheuristiques les plus connues, elle a été introduite par Glover (Glover, 1986). Elle utilise un historique de manière à interdire à l'algorithme de revenir sur ses pas. Cet historique se traduit par la présence d'une liste dite tabou qui garde une trace des dernières solutions visitées. Ainsi l'algorithme ne pourra plus explorer ces solutions (du moins à court terme, tout dépend de la taille de la liste tabou).

L'algorithme fonctionne comme suit. Initialement la liste tabou est vide et on génère une solution *S* à l'aide d'une heuristique quelconque ; *S* devient la solution courante. A chaque itération, on choisit le meilleur voisin *S'* de *S* qui n'est pas déjà dans la liste tabou, *S'* devient la solution courante *S* et est ajoutée à la liste tabou. Si la taille de la liste tabou dépasse la taille maximale autorisée, on supprime de cette liste l'élément le plus ancien (stratégie *FIFO - First In First Out*). Notons que la liste tabou permet d'éviter les cycles en interdisant de choisir une solution dans le voisinage de la solution courante qui aurait déjà était explorée. En outre, en pratique, on ne stocke pas les solutions dans leur intégralité (trop coûteux en temps et en espace) mais seulement une signature de ces solutions.

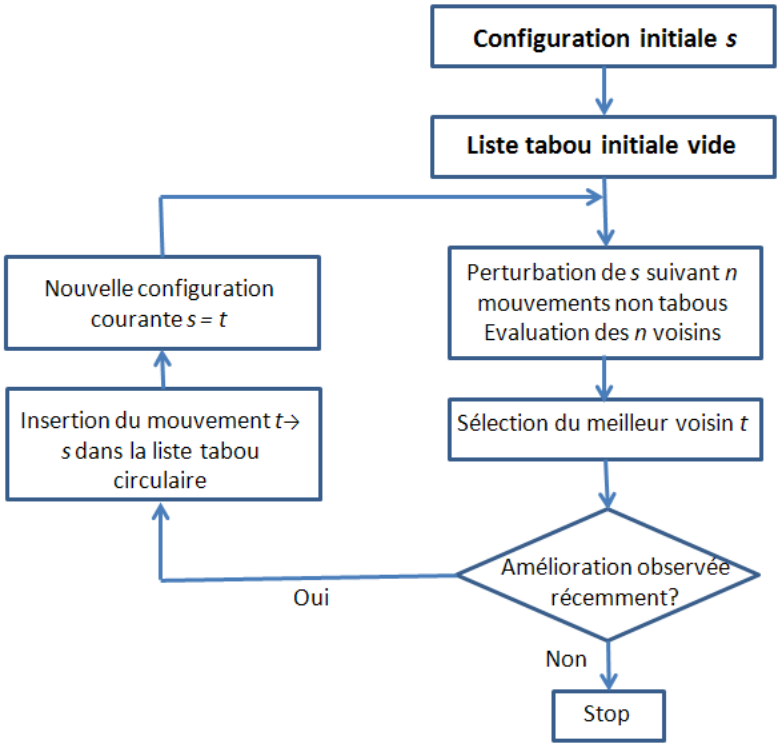

Figure II. 2 Organigramme de l'algorithme Tabou (Siarry, 2002)

II.2.2.4 Les algorithmes évolutionnaires

Les algorithmes évolutionnaires sont inspirés du domaine de la biologie : ils ont été introduits dans les années 1950 (Fraser, 1957). Les techniques utilisées reproduisent le schéma d'évolution des espèces et en adoptent le vocabulaire. Les solutions sont appelées individus et l'algorithme traite simultanément plusieurs individus. L'ensemble de ces individus est appelé population et évolue à chaque itération de l'algorithme. La population relative à une itération donnée s'appelle génération, on obtient donc une nouvelle génération à chaque itération. Les individus qui servent à produire la nouvelle génération sont appelés parents et les individus résultants sont appelés enfants ou fils.

Le principe d'un algorithme évolutionnaire est simple. En effet, il s'agit de considérer un ensemble de *N* points, choisis à priori au hasard, dans un espace de recherche. Cet ensemble constitue la population initiale. Chaque individu *x* de la population ainsi définie possède une certaine performance, qui mesure son degré d'adaptation à l'objectif visé. Par exemple, dans le cas de la minimisation d'une fonction objectif ***f***, l'individu *x* est d'autant plus performant que *f(x)* est plus petit. Le principe de base d'un algorithme évolutionnaire revient à faire évoluer de manière progressive, par générations successives d'individus, la composition de la population tout en maintenant sa taille constante. Au fil des générations, l'objectif principal visé consiste à améliorer la performance globale des individus (Siarry, 2002).

Plusieurs approches d'algorithmes évolutionnaires ont été étudiées:

- Les stratégies d'évolution (*evolution strategies* - *ES*), utilisées pour résoudre des problèmes d'optimisation continue ;
- La programmation évolutionnaire (*evolutionary programming* - *EP*), conçue pour faire évoluer des automates à états finis, l'idée étant de créer une « intelligence artificielle » ;
- Les algorithmes génétiques (*genetic algorithms* - *GA*), utilisés pour résoudre des problèmes d'optimisation combinatoire, ces derniers étant certainement les plus populaires.

Exemple : algorithme génétique

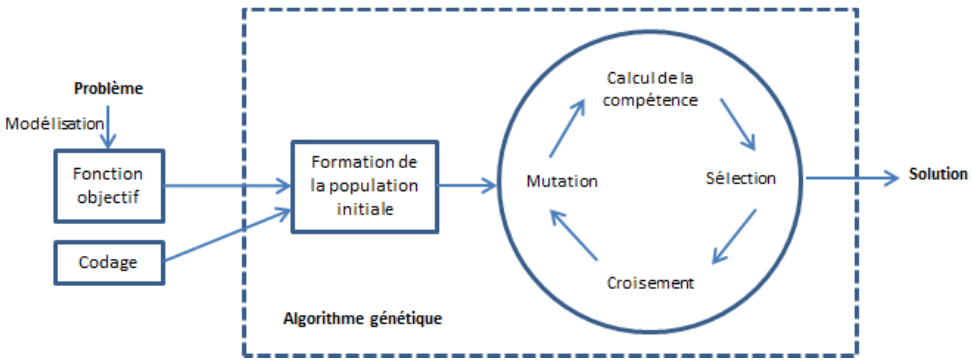

Figure II. 3 Résolution d'un problème par algorithme génétique (Siarry, 2002)

II.2.3 Méthodes exactes

Nous présentons dans la section suivante deux méthodes exactes: l'exploration arborescente et la programmation linéaires qui sont les deux méthodes les plus utilisées dans la résolution des problèmes d'optimisation combinatoire.

II.2.3.1 La méthode par séparation-évaluation (branch & bound)

La méthode par séparation-évaluation consiste à générer progressivement tout l'espace de recherche (que l'on représente comme les feuilles d'un arbre). Pour gagner du temps, nous évitons de générer toutes les feuilles, leurs nombre étant exponentiel par rapport à la taille de

l'instance. Le principe est de générer l'arbre progressivement en essayant de commencer par les sous-arbres les plus prometteurs et surtout en n'explorant pas jusqu'aux feuilles un sous-arbre dont on sait qu'il ne peut pas contenir la solution optimale.

Le fonctionnement général de cet algorithme est divisé en trois étapes :

- La séparation : Le terme séparation fait référence à la stratégie d'énumération de toutes les solutions. Cette stratégie consiste à construire l'arbre d'exploration. Pour cela on a besoin de déterminer l'opérateur de construction qui va permettre d'ajouter un élément à la solution partielle et la stratégie de parcours de l'arbre qui détermine l'ordre d'exploration des nœuds.

Au fur et à mesure de l'exploration, on obtient des informations qui permettent, grâce à la phase d'évaluation, d'éviter l'exploration de certaines parties de l'arbre.

- L'évaluation : Cette phase de l'algorithme permet d'exclure, au cours du processus d'exploration, certaines branches de l'arbre dont on sait qu'elles ne peuvent pas contenir la solution optimale. Pour cela nous devons connaitre la valeur de la borne supérieure sur la solution optimale et la fonction qui va permettre d'évaluer la solution partielle ;

- Le parcours de l'arbre : Il existe deux stratégies de parcours. Nous pouvons parcourir l'arbre en profondeur, c'est à dire en descendant dans les branches jusqu'à trouver une solution (feuille) ou une solution que l'on peut éliminer. Dans ce dernier cas, on remonte dans la branche pour redescendre dans une autre direction. On peut choisir de parcourir les branches dans l'ordre de l'arbre (de gauche à droite) ou alors en commençant par les nœuds de coût les plus faibles dans un même niveau (qui semblent les plus prometteurs). La deuxième stratégie consiste à parcourir l'arbre en largeur. Dans ce cas nous explorons les solutions niveau par niveau. Cette stratégie est rarement utilisée en pratique car elle ne permet pas un filtrage efficace et elle est très coûteuse en place mémoire.

II.2.3.2 La programmation linéaire

On appelle programme (ou modèle) linéaire un modèle mathématique représentant un problème d'optimisation, dans lequel on cherche à optimiser une fonction objectif linéaire sous un certain nombre de contraintes. Ces contraintes sont modélisées par des équations ou inéquations linéaires. Il existe des méthodes qui assurent la résolution exacte d'un tel programme. Une des méthodes les plus connues est la méthode du simplex (de son inventeur G.B. Dantzig). En théorie, cette méthode a une complexité non polynomiale. Cependant, il s'avère qu'elle est très efficace en pratique. Comme toutes les techniques de résolution exacte, les temps de calcul peuvent vite devenir très grands. Il est donc courant d'utiliser des techniques supplémentaires comme la génération de colonnes ou le *branch and cut* (coupes et

branchements) pour les problèmes de grande taille. Des librairies d'optimisation comme CPLEX (d'IMB ILOG) implémentent ces méthodes.

Dans cette section du chapitre, nous nous sommes intéressés à des problèmes d'optimisation combinatoire et nous avons passé en revue différentes approches de résolution. Nous avons passé en revue les deux grandes approches de résolution existantes. D'une part, les heuristiques et métaheuristiques qui fournissent des algorithmes de résolution approchée sans garantir la qualité de la solution. D'autre part, les méthodes exactes telles que l'exploration arborescente ou la programmation linéaire, qui permettent de calculer des solutions optimales mais avec des temps de calcul pouvant être prohibitifs. Le schéma de la figure II.4 résume la classification des méthodes de résolution.

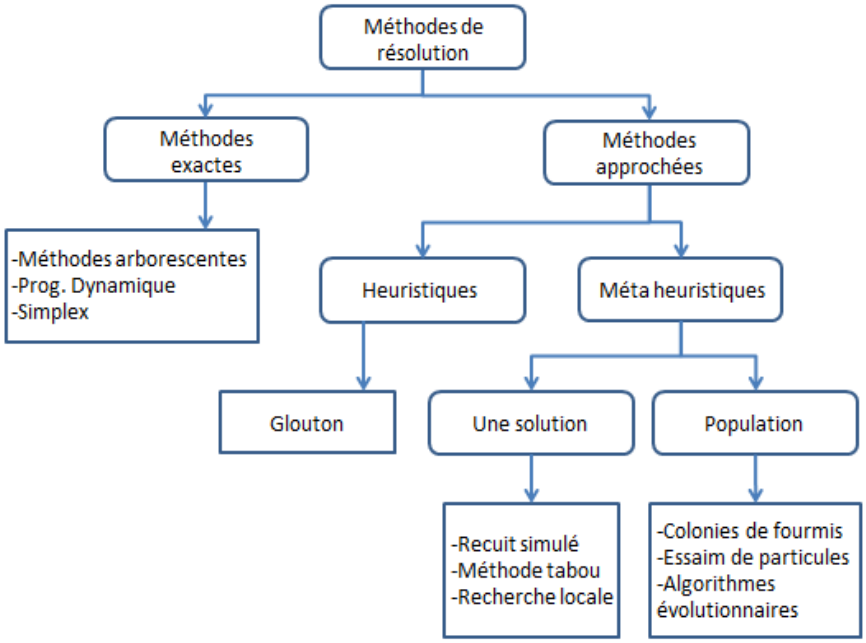

Figure II. 4 Classification des méthodes de résolution

Dans la section suivante, nous nous intéressons au paradigme multi-agent que nous considérons bien adapté pour la mise en œuvre et la modélisation des problèmes d'optimisation pour les systèmes distribués.

II.3 Les systèmes multi-agents

Dans cette section, nous commençons d'abord par définir les différentes notions liées à cette méthode, particulièrement : l'éco-résolution, la coopération, la négociation, et la planification qui constituent les concepts clés de notre chaîne logistique de gestion de crise. Ensuite nous abordons les différents aspects de la formalisation des systèmes multi-agent. Enfin, nous justifions notre intérêt pour l'adoption de ce paradigme en citant nos différents arguments.

II.3.1 Quelques concepts

Les systèmes multi-agents forment un paradigme pour la conception des systèmes complexes et proposent des outils pour les analyser, les concevoir et les implanter. Théoriquement, un Système Multi-Agent (SMA) est un ensemble d'entités (agents logiciels ou humains) qui interagissent dans un environnement pour résoudre des problèmes qui dépassent les capacités ou les connaissances individuelles de chaque agent. Grâce à sa nature autonome, réactive et/ou proactive, le paradigme agent est adopté dans des systèmes dynamiques et en temps réel. Il est intéressant d'observer la place qu'occupent les systèmes multi-agent dans l'évolution des paradigmes de l'informatique selon (Jamont, 2005). La figure II.5 suivante illustre cette place.

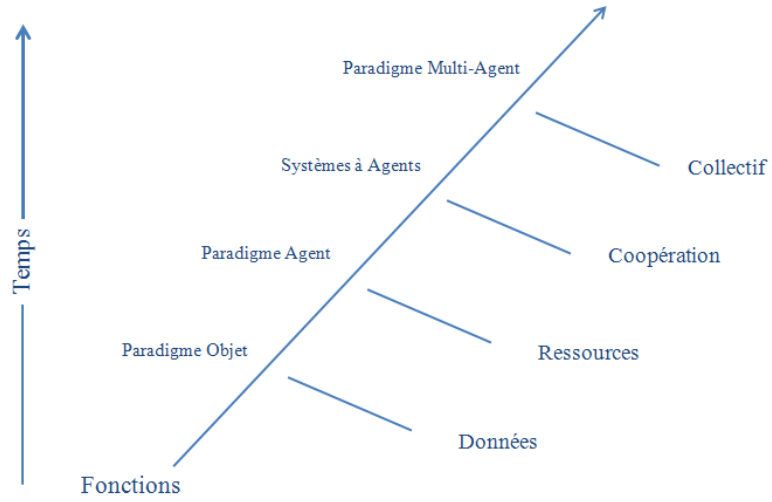

Figure II. 5 Evolution des paradigmes de l'informatique

II.3.1.1 La notion d'agent

Parmi les nombreuses définitions du terme « agent », certaines semblent faire l'unanimité au sein de la communauté multi-agent. Ferber (Ferber, 1995) définit un agent comme une entité physique ou virtuelle :

– qui est capable d'agir dans un environnement,

– qui peut communiquer directement avec d'autres agents,

– qui est mue par un ensemble de tendances (sous la forme d'objectifs, individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser),

– qui possède des ressources propres,

– qui est capable de percevoir (mais de manière limitée) son environnement, qui ne dispose que d'une représentation partielle de cet environnement (et éventuellement aucune),

– qui possède des compétences et offre des services,

– qui peut éventuellement se reproduire,

- dont le comportement tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit.

Une définition plus commune, inspirée des travaux de Jennings (Jennings et al., 1998), englobe ces notions et définit un agent comme étant « un système informatique doué de raisonnement et capable de communiquer avec ses semblables et qui agit sur son environnement d'une façon autonome et en fonction de sa perception pour atteindre les objectifs pour lesquels il a été conçu ». Cette définition met l'accent, non seulement sur la capacité de l'agent à raisonner, mais aussi à agir. En effet, le concept d'agent repose sur la notion fondamentale de l'accomplissement des actions qui vont agir sur l'environnement et le modifier. Il en découle de cette définition un cycle de vie pour l'agent basé sur trois grandes phases : La perception de l'environnement, la prise de décision et l'exécution. L'agent utilise ses connaissances et ses comportements pour construire sa perception de l'environnement qui l'entoure et le reproduire. Ensuite vient l'étape de la prise de décision ou encore « délibération » qui consiste à décider de la prochaine action à exécuter. Pour ce faire, l'agent use de ses raisonnements et de son « intelligence » qui caractérisent sa capacité à être autonome. Enfin, l'agent effectue concrètement l'action choisie lors de l'étape précédente.

Une autre définition est apportée par Wooldridge (Wooldridge et al., 1995), un agent est un système informatique capable d'agir de manière autonome et flexible dans son environnement. La flexibilité d'un agent consiste en les propriétés suivantes :

– Réactivité : elle concerne l'aptitude d'un agent à réagir aux modifications survenant dans son environnement et à adapter son fonctionnement et comportement (actions et ordre dans lequel elles sont entreprises) en fonction des changements perçus.

– Pro-activité : elle concerne l'aptitude d'un agent à agir de sa propre initiative pour essayer d'atteindre les buts et objectifs qu'il s'est fixé. Son comportement n'est donc pas uniquement dirigé par des événements mais il est tout à fait capable de générer et satisfaire des buts.

– Capacités sociales : un système social est capable d'interagir ou coopérer avec d'autres systèmes. Les agents communiquent directement entre eux pour pouvoir échanger des informations. Ils interagissent via des langages de communication spécifiques et convenus.

II.3.1.2 Typologie d'agents

La différence dans la typologie des agents découle de la différence comportementale par rapport à la manière et la rapidité d'exécution des actions qui leur sont spécifiques (Bradshaw, 1997) (Strugeon, 1995). Trois principaux agents sont distingués dans la littérature :

- Agents cognitifs : Ces agents proviennent d'une métaphore du modèle humain. Ils possèdent une représentation explicite de leur environnement, des autres agents et d'eux-mêmes. Ils sont dotés de capacités cognitives : anticiper, raisonner, mémoriser, planifier, communiquer, etc. Les actions qu'ils entreprennent sont généralement le fruit d'un comportement réfléchi basé sur leurs connaissances de l'environnement. Le travail le plus représentatif de cette famille d'agent porte sur le modèle *Believe Desir Intention BDI* (Rao & Georgeff, 1995).

- Agents réactifs : très rapides, leurs actions sont assimilées à des réflexes. Ils fonctionnent selon un modèle stimuli/réponse. En effet, dès qu'ils aperçoivent un changement dans leur environnement, ils réagissent par une action programmée. Pour cela, ces agents sont constamment en état de veille attendant un changement probable survenant dans l'environnement. Contrairement aux agents cognitifs, les agents réactifs n'ont pas de représentation explicite de leur environnement. L'exemple le plus célèbre est celui de la fourmilière étudiée par Alexis Drogoul (Drogoul, 2000).

- Agents hybrides : Ces agents conjuguent la rapidité de réponse des agents réactifs avec les capacités de raisonnement des agents cognitifs. Il s'agit de ce fait d'un compromis autonomie/coopération et efficacité/complexité.

II.3.1.3 Les système multi-agents

A partir des définitions précédentes de la notion d'agent, nous pouvons considérer un système multi-agent comme étant un ensemble homogène ou hétérogène d'agents situés dans un même environnement et qui interagissent. Ces agents communiquent et collaborent pour achever des objectifs individuels ou collectifs, ou pour résoudre des problèmes qui dépassent les capacités ou les connaissances individuelles de chacun. Ferber (Ferber, 1995) définit un système multi-agent de la manière suivante :

" Un système multi-agent est un système composé des éléments suivants :

– Un environnement qui est un espace disposant généralement d'une métrique.

– Un ensemble d'objets situés dans l'espace, ils sont passifs, ils peuvent être perçus, détruits, créés et modifiés par les agents.

– Un ensemble d’agents qui sont les entités actives du système.

– Un ensemble de relations qui unissent les objets entre eux.

– Un ensemble d’opérations permettant aux agents de percevoir, de détruire, de créer, de transformer et de manipuler les objets.

– Un ensemble d’opérateurs chargés de représenter l’application de ces opérations et la réaction du monde à cette tentative de modification (les lois de l’univers)."

Selon la vision de Demazeau (Demazeau, 1995) :

SMA = Agent +Environnement + Interaction + Organisation.

Cette décomposition d’un SMA est dite décomposition Voyelles qui identifie un axe Agent, un axe Environnement, des Interactions et une structure Organisationnelle explicite ou non.

L’interaction représente l’une des plus importantes caractéristiques d’un SMA. Elle est définie par Doniec (Doniec, 2006) comme étant une relation dynamique instaurée entre plusieurs agents du fait de leurs actions combinées et réciproques. L’interaction peut avoir plusieurs formes : la coopération, la coordination, la négociation, la collaboration, etc. Toutefois, elle est essentiellement représentée par la coordination, qui fait référence à tous les processus utilisés dans la prise de décision et au comportement global d’un ensemble d’agents.

II.3.2 Les mécanismes d’interactions

Les interactions, telles que définies par Ferber dans (Ferber, 1995), correspondent à une mise en relation directe et dynamique de deux agents ou plus et ce par le biais d’un ensemble d’actions réciproques. Weiss (Weiss, 2000) propose la modélisation donnée par la figure II.6 suivante afin de schématiser l’ensemble des interactions possibles entre les agents. Il définit la coordination en faisant référence aux ressources partagées car les agents doivent coordonner leurs actions dans l’environnement qu’ils partagent afin d’éviter les conflits de ressources.

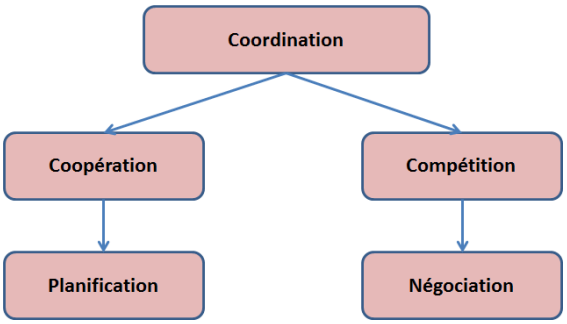

Figure II. 6 Les interactions sous leurs différentes formes

La coordination est définie au sens général par Doniec (Doniec, 2006), comme étant l'ensemble des actions permettant une production et/ou une organisation en vue d'atteindre un objectif déterminé. Dans ce sens, soit les agents s'entre-aident et donc coopèrent pour essayer d'atteindre l'objectif global en évitant les situations de conflits, soit ils se retrouvent dans une situation de compétition.

II.3.2.1 La coopération

Une situation coopérative est une situation où les agents collaborent pour réaliser un but commun. Ce but peut être soit un but partagé par tous les agents, soit un but mis en avance par le concepteur du SMA. Les agents peuvent être alors associés à un ou plusieurs buts. La coopération apparaît quand les actions de chaque agent satisfont au moins l'une des conditions suivantes: (1) les agents ont un but commun à tous et leurs actions tendent à réaliser ce but, (2) les agents effectuent les actions qui réalisent non seulement leurs propres buts mais aussi ceux des autres. Dans ce cas, les agents ne sont pas dans une situation de concurrence, ils essaient donc de s'accommoder sans se déranger. Les agents peuvent coopérer pour partager les tâches et les ressources. C'est une méthode qui permet la répartition du travail.

II.3.2.2 La planification

Bond et Gasser (Bond & Gasser, 1988) nous font remarquer que l'orientation des comportements des agents vers des buts communs permet d'obtenir une plus grande coordination. (Boissier , 2000) présente plusieurs techniques de planification comme la planification centralisée pour des plans distribués, la planification distribuée pour des plans centralisés et la planification distribuée pour des plans distribués :

a) Planification centralisée pour des plans distribués : Ce type de planification se focalise sur le contrôle et la coordination par plusieurs agents dans des environnements partagés. En planification multi-agent centralisée, un agent est responsable de la création du plan qui spécifie les actions planifiées pour tous les agents concernés. Le principe est le suivant :

- Génération de plans à ordre partiel,
- Décomposer le plan en sous-plans,
- Insérer des actions de synchronisation dans les sous-plans en introduisant des commandes de communication,
- Allouer les sous-plans aux agents,
- Initier l'exécution du plan.

Ce type de planification présente l'avantage de faciliter la résolution de conflits et de converger vers une solution globale. Cependant la mise en place nécessite la centralisation du contrôle au niveau d'un seul agent.

b) Planification distribuée pour des plans centralisés : Dans une approche de planification distribuée, les activités de planification sont réparties au sein d'un groupe d'agents. Cette approche est utilisée quand un seul agent ne peut pas avoir une vue globale des activités du groupe. Le processus de planification est alors distribué entre différents agents spécialistes qui coopèrent et communiquent en partageant des objectifs et des représentations pour former un plan cohérent. Les spécialistes génèrent leurs portions de sous-plans, et les plans partiels sont ensuite synchronisés en un seul plan (Boissier , 2000). Cette technique présente l'avantage de ne pas centraliser le contrôle au niveau d'un seul agent. Cependant elle engendre des coûts de communication élevés et sans garantie de convergence, puisque les agents peuvent avoir des incompatibilités au niveau des buts et intentions.

c) Planification distribuée pour des plans distribués : Pour ce type de planification, le processus de synthèse d'un plan et son exécution sont distribués. Chaque agent produit un plan, soit indépendamment des autres (agent-driven), soit en étant dirigé par un but commun avec les autres (goal-driven). Les agents peuvent se communiquer leurs plans.

II.3.2.3 La négociation

Contrairement à la coopération qui suppose la sociabilité des agents, la négociation correspond à la coordination en univers compétitif entre agents. La négociation est un mécanisme puissant pour gérer les dépendances inter-agents ou le processus par lequel un groupe d'agents arrive à une décision mutuelle acceptable sur un sujet donné.

Dans un SMA, un objet est considéré comme un sujet de négociation entre un groupe d'agents, s'il se retrouve dans une situation de désaccord pour la réalisation d'un objectif commun. Ainsi, la négociation est la forme de contribution la plus appropriée pour la résolution de conflits car elle garantit un pouvoir équitable des agents, ce qui permet la

décentralisation du contrôle. Le principe de base pour mettre en place un mécanisme de négociation est de faire, pour les agents négociants, des propositions, des choix, et des concessions d'offres, pour éventuellement atteindre une décision commune acceptable. Dans les systèmes multi agents, l'objectif est de construire des agents négociants avec l'intervention humaine la plus réduite possible.

Les agents engagés dans un processus de négociation doivent prendre certaines initiatives qui peuvent conduire à un accord entre les différentes parties. De ce fait, trois grands mécanismes de négociation existent dans la littérature :

• La négociation par compromis : dans ce type de négociation, chacune des parties relâche les contraintes qu'elle juge moins importantes afin d'augmenter les chances d'atteindre le but global,

• La négociation intégrante : dans ce type de négociation, chacune des parties est amenée à changer ses objectifs selon le nouveau tour de négociation courant, en cherchant les buts profonds,

• La formation de coalitions: il s'agit de partitionner le groupe d'agents en conflit en sous-groupes appelés coalitions. Une coalition représente une organisation d'agents à court terme basée sur des engagements spécifiques et contextuels, ce qui permet aux agents de coexister tout en bénéficiant de leurs compétences respectives (Vauvert et al., 2000).

II.3.3 Apport des SMA

Tels que longuement décrits dans les sections précédentes ainsi que dans une multitude de travaux de recherche (Feki, 2010) (Zgaya, 2007) (Kammoun, 2007), les SMA présentent de multiples avantages. Ces derniers ont fait leur grand succès dans plusieurs domaines et le recours des chercheurs et industriels aux plateformes multi-agents pour la mise en place de systèmes performants et efficaces. Parmi leurs points forts, nous pouvons noter essentiellement :

- Leur capacité d'adaptation à n'importe quel type de problème.

- L'intégration et la gestion aisée d'une quantité importante d'informations distribuées et hétérogènes faisant évoluer la littérature vers le génie logiciel distribué.

- La personnalisation des services à l'usager de par l'intégration d'agents assistants l'utilisateur dans son usage du système. Ces agents sont appelés *Agents Interface*. De par cette

pratique, les systèmes d'information s'en trouvent ainsi personnifiés s'adaptant aux besoins des logisticiens.

- La prise de décision collective à travers des mécanismes de coordination et collaboration assurées de par la communication, échanges et interactions entre les agents. Dans ce cadre, les *Agents Collaborants* sont connus pour leur efficacité à interconnecter plusieurs systèmes ou résoudre des problèmes de type distribué. Ils associent pour cela leurs compétences individuelles pour mener à bien leurs tâches dans le but de réaliser un objectif collectif dont la complexité dépasse leurs aptitudes individuelles. Ce domaine d'application est souvent appelé *Intelligence Artificielle Distribuée* (*IAD*).

Le domaine des systèmes multi-agents (SMA) propose donc des méthodologies, des techniques et des outils pour répondre à différents problèmes que l'on peut décliner selon trois aspects :

- Le développement de systèmes informatiques décentralisés où l'approche SMA permet l'intégration flexible et la coopération de logiciels et services autonomes,
- La résolution de manière distribuée d'un problème qui se pose globalement à la collectivité d'agents,
- La simulation de phénomènes complexes où la modélisation multi-agent apporte un cadre conceptuel permettant la représentation et la simulation de systèmes faisant intervenir différentes entités en interaction.

II.3.4 Applications des SMA

Plusieurs publications concernant les applications des systèmes multi-agents dans l'industrie (Chaib-Draa, 95) (Kouiss & al., 95) (Shen & al., 2006) ont fait leurs apparition. Cette émergence s'explique par l'aptitude des SMA à modéliser des systèmes de plus en plus complexes. Parmi les premières applications développées à base d'agents communicants, on trouve une application pour le contrôle du trafic aérien (Cammarata & al., 1988). Dans cette application, les stratégies de coopération des SMA ont été exploitées pour résoudre les conflits entre les plans d'un groupe d'agents. Chaque agent, représentant un avion, cherche à bâtir un plan de vol qui doit garantir une distance sécuritaire de chaque autre agent. Dans le cas où les agents sont en conflit, ils doivent choisir entre l'agent le plus apte à élaborer un nouveau plan de vol sans engendrer de nouvelles situations conflictuelles. Ce choix se base sur la quantité d'information dont dispose chaque agent et sur le nombre de contraintes. En

effet, il revient à l'agent le mieux informé de réaliser le nouveau plan de vol qui résout le conflit et à l'agent le moins contraint de l'exécuter.

Malgré leur récente expansion, les SMA touchent une grande variété de domaines d'applications. En effet, ils sont à l'intersection de plusieurs domaines scientifiques (Chaib-Draa et al., 2001): informatique répartie et génie logiciel, intelligence artificielle et vie artificielle, sociologie, psychologie sociale, sciences cognitives et bien d'autres. Nous retrouvons ainsi les SMA à la base des systèmes distribués, des interfaces homme-machines, des bases de données et bases de connaissances distribuées coopératives, de la robotique cognitive et coopération entre robots et des applications distribuées comme le web, l'internet, le contrôle de trafic routier, le contrôle aérien, les réseaux d'énergie, etc.

Dans la section suivante, nous essayons de mettre en évidence et de prouver l'intérêt de l'alliance entre les SMA et l'optimisation pour modéliser et optimiser, de manière conjointe, des modèles complexes en logistique. Nous allons, de ce fait, dresser un état de l'art sur les approches agent en optimisation. Nous choisissons de présenter les méthodes les plus souvent référencées dans la littérature concernant le mariage entre SMA et optimisation.

II.4 L'alliance entre les SMA et l'Optimisation

Les systèmes multi agents et l'optimisation offrent par leur combinaison, une approche de résolution qui permet de contrer le problème de complexité des systèmes réels. En effet, l'association des travaux théoriques réalisés dans le domaine de l'optimisation avec les systèmes d'aide à la décision, à base d'agents communicants, facilite la préparation rationnelle et l'automatisation de décisions humaines importantes et complexes. Deux aspects essentiels rapprochent les méthodes d'optimisation des SMA : le premier est la distribution du calcul, le second est lié aux problèmes d'adaptation et d'auto-adaptation de la recherche. En effet, le choix d'une bonne stratégie de recherche (ou le choix d'un bon paramétrage) et la résolution du problème peuvent être réalisés dynamiquement et de façon distribuée. Le problème de sélection d'actions à réaliser en fonction de la situation pour un agent donné, peut alors être assimilé à un problème d'apprentissage, dans un objectif général de maximisation d'un gain (Meignan, 2008).

II.4.1 Les SMA pour une résolution collective d'un problème d'optimisation

Au sens de cette approche, un agent est associé à la construction ou à l'amélioration d'une solution unique d'un problème d'optimisation. Les communications et interactions entre agents visent à échanger des informations à propos des zones prometteuses de l'espace de

recherche, de manière à rendre l'exploration plus efficace. Par exemple, pour l'optimisation par colonies de fourmis, les agents échangent les informations et coopèrent par le biais de l'échange de la matrice de phéromone. Cette situation de coopération se nomme la stigmergie (Parunak, 2003). L'intérêt de l'utilisation du paradigme multi-agent pour les approches d'optimisation est tout d'abord de pouvoir décrire de manière naturelle le fonctionnement des méthodes d'optimisation, ce qui en facilite la conception. Donc, dans un premier temps, les SMA offrent un support informatique performant et flexible pour la modélisation et la mise en œuvre des problèmes d'optimisation. Ensuite, en termes de résolution, la distribution du calcul permet une exploration plus efficace de l'espace de recherche tout en permettant une mise en œuvre parallèle possible des algorithmes mis en place. Les systèmes multi-agents semblent donc adaptés pour la résolution de divers types de problèmes combinatoires. Cette alliance permet une application simultanée de plusieurs algorithmes de recherche locale, et nous offre la possibilité d'explorer un grand nombre de solutions, en prenant différentes directions définies par divers fonctions de voisinage considérées à la fois, et ainsi accélérer la recherche d'une bonne solution. En effet, les propriétés sociales qui caractérisent les systèmes multi-agents, telles que la communication et la coopération, garantissent la visite d'un grand nombre de solutions, chose qui ne s'avère pas possible si chaque heuristique est utilisée indépendamment des autres.

Nous nous basons alors sur les principes des systèmes multi-agents, qui nous permettent d'avoir une collaboration entre les différentes méthodes et heuristiques. Notons que l'utilisation de ces systèmes, pour la résolution de certains problèmes industriels, présente des avantages certains dus à la fois, à l'exécution parallèle de plusieurs procédures et aussi à la présence de collaboration entre la totalité de ses composantes. Cette collaboration, assurée par une communication permanente entre les agents, a pour rôle de donner à chacun d'entre eux la possibilité d'avoir une estimation exacte de l'évolution actuelle du système, et par conséquent une bonne orientation aux procédures. En d'autres termes, la possibilité qu'offre cette démarche de lancer simultanément les différents agents, exécutant chacun de son côté la procédure qui lui est associée, permet au système d'augmenter le nombre de bonnes solutions calculées. Ces solutions pourront alors servir de solutions de départ pour de prochaines itérations.

Le concept d'agent est souvent explicitement mentionné dans les métaheuristiques distribuées. C'est le cas par exemple de la méthode Co-Search (Talbi & Bachelet, 2004). Cette méthode est basée sur l'utilisation de trois agents complémentaires, chacun d'eux ayant

un rôle bien défini et combinant des structures de voisinage de différentes portées notamment via un algorithme génétique. Ces trois agents évoluent en parallèle et coopèrent via une mémoire centrale et partagée. Pendant la recherche, l'ensemble des connaissances acquises par les trois agents est regroupé dans cette mémoire centrale. Le premier agent, appelé *Searching Agent*, utilise une fonction d'amélioration basée sur le voisinage. En début d'exécution, le *Searching Agent* utilise la Recherche Tabou pour améliorer la solution initiale. Le second agent est appelé *Diversifying agent*. Il est chargé de fournir de nouvelles solutions dans les zones non explorées de l'espace de recherche, en exploitant les informations des régions déjà visitées. Pour ceci, il utilise un algorithme génétique. Le dernier agent est appelé *Intensifying Agent* et a pour rôle de générer de nouvelles solutions à partir de zones prometteuses déjà explorées, en utilisant un opérateur de perturbation. La méthode *Co-Search* permet de visiter de nouvelles régions et d'explorer de manière plus précise des régions déjà visitées. C'est une méthode qui semble très intéressante du point de vue de la combinaison de plusieurs méthodes différentes en parallèle et donc de faire coopérer plusieurs stratégies de recherche complémentaires.

II.4.2 Les SMA, une solution adaptée au problème d'apprentissage

Un autre avantage de l'alliance entre les SMA et des méthodes d'optimisation est directement lié aux problématiques d'adaptation et d'apprentissage des stratégies de recherche. En effet, le paradigme multi-agent offre la possibilité de choisir la bonne stratégie de recherche ou le bon paramétrage dynamiquement et en temps réel, sous la forme d'un problème d'apprentissage. Plusieurs travaux de recherche ont traité les problématiques d'adaptation. Cette problématique associée aux systèmes multi-agents est souvent assimilé au problème d'apprentissage par renforcement (Burke et al, 2003). Un paradigme classique pour présenter les problèmes d'apprentissage par renforcement consiste à considérer un agent autonome, plongé au sein d'un environnement, et qui doit prendre des décisions en fonction de son état courant. En retour, l'environnement procure à l'agent une récompense (un gain à maximiser).

Plusieurs familles d'algorithmes multi-agents mettant en œuvre les mécanismes d'adaptation ont vu le jour. La plus grande source d'inspiration reste cependant l'observation de la nature. Les algorithmes biomimétiques s'inspirant du comportement social observé chez les fourmis en sont un bon exemple.

Un autre exemple d'apprentissage des algorithmes génétiques à base d'agents a également été mis à contribution dans (Vallin et Coello, 2003) pour la recherche d'information sur Internet.

Le but est de créer un agent chargé d'assister les utilisateurs dans leurs problèmes de recherche d'information. Cependant, ce système a la particularité d'être conçu à la manière d'un outil de veille stratégique : il est chargé de scruter en permanence une zone prédéfinie d'Internet afin d'y détecter l'apparition de documents pertinents pour un utilisateur. L'originalité de ce système réside dans sa manière de récupérer les souhaits de l'utilisateur et de les adapter au fur et à mesure de son utilisation.

II.4.3 Besoins liés à la résolution distribuée des problèmes d'optimisation

Les propriétés souhaitables pour un algorithme d'optimisation exécuté au sein d'une architecture distribuée tel un système multi-agent sont la distribution, la décentralisation, la robustesse aux pannes, l'asynchronisation des échanges de messages, l'autonomie des agents, le fonctionnement dans un système ouvert, etc.

- *La distribution* implique la répartition de l'exécution de l'algorithme sur différents agents. Concrètement cela veut dire que chaque agent doit être doté d'un processeur et d'un algorithme local s'exécutant sur celui-ci, participant au calcul de l'algorithme global. L'exécution de l'algorithme globale se fait donc de manière concurrente sur plusieurs machines. L'état de l'algorithme global est défini comme l'union des états des algorithmes locaux.

- *La décentralisation* exige qu'aucun agent ne soit distinguable des autres du point de vue de l'algorithme, en d'autres termes, du point de vue des connaissances et capacités nécessaires au bon déroulement de l'algorithme. Autrement dit, nous dirons que l'algorithme est centralisé si un seul agent détient une information sans laquelle l'exécution de l'algorithme devient impossible.

- *La robustesse aux pannes* sous-entend que l'algorithme puisse s'exécuter quel que soit le nombre d'agents tombant aux pannes, en répliquant les ressources critiques du système.

- *Les pannes du réseau de communication* peuvent retarder l'arrivée des messages ou intervertir leurs ordres d'arrivée par rapport à leurs ordres d'envoi. De ce fait, un échange de message synchrone pourrait faire en sorte qu'un agent attende indéfiniment l'arrivée d'un message. Pour faire face à ce genre de situations, et afin de détecter qu'un agent est tombé en panne, des détecteurs de défaillance sont mis en place. Mais ces détecteurs fonctionnant eux-mêmes à base d'échange de messages, nécessitent des performances plus fortes qui relèvent du domaine des réseaux et non des systèmes multi-agents.

- *La synchronisation des algorithmes* est une propriété nécessaire qui peut s'avérer problématique car elle sous-entend que plusieurs agents (ou tous), réalisent simultanément une transition dans les séquences d'exécution de leur algorithme local. Pour faire ceci, nous devons posséder une information globale qui concerne tous les agents. Cependant une telle information est considérée centralisée car même si elle est partagée par tous les agents, elle nécessite la collecte d'une information sur l'état global du système. Il est donc préférable de fonctionner de manière asynchrone, car cela implique que la procédure n'est pas centralisée et donc plus robuste aux pannes.

II.5 Quelques systèmes multi-agents pour l'optimisation de la logistique de gestion de crise

Nous nous intéressons, en particulier, aux applications des systèmes multi-agents dans le domaine de la logistique de défense. Nous présentons brièvement, dans cette section, trois applications, compte tenu des publications limitées pour des contraintes de confidentialité.

II.5.1 Nexus : Auto-organisation à base d'agents pour le soutien en combat

Le système Nexus est un middleware Peer-to-Peer (P2P) orienté agents qui permet de créer une architecture orientée service hautement distribuée. Nexus propose une combinaison structurée d'un réseau de recouvrement P2P et de services autonomes, et offre un puissant moyen pour soutenir les opérations en temps réel dans la sécurité ou la défense. Nous présentons, dans ce qui suit, une vue d'ensemble sur ce système.

II.5.1.1 Approche

La première phase du projet Nexus a démontré la valeur d'un middleware basé agent P2P pour la découverte et la fusion de services. Il est basé sur trois paradigmes principaux : l'informatique P2P, les agents autonomes et les architectures orientées services. Les implémentations existantes d'architecture orientée services, telles qu'elles sont appliquées dans le domaine civil, souffrent de plusieurs limitations qui les rendent impropres pour les environnements volatiles. En effet, des facteurs comme l'orchestration de processus, les workflows spécifiés manuellement et les services centralisés conduisent à des applications réseau fragiles, non adaptatifs et difficiles à entretenir. L'objectif est de développer une architecture orientée services orientée agents, qui répond aux exigences de fiabilité rigoureuse du domaine de la défense et accueille les besoins du réseau des applications centrées sur la fusion de l'information. En particulier, les fonctionnalités suivantes sont en cours d'élaboration :

- Prestation de services homogènes et fiables dans les environnements volatiles.
- Hiérarchisation des demandes et répartition des charges.
- Résilience à la volatilité de l'infrastructure réseau sous-jacent: en adoptant une architecture *peer-to-peer*.
- Services décentralisés où les ressources en réseau sont présentées en fonction de leurs propriétés et des informations reçues en temps réel concernant leurs attributs.
- Sélection sémantique et adaptative de services.
- La surveillance proactive et de la substitution du service automatisé.
- Filtrage des services d'information en fonction de leur pertinence sémantique pour l'utilisateur ainsi que l'imposition d'une certaine structure à la couche de messagerie du middleware.

Afin d'offrir la résilience nécessaire, Nexus adopte une approche entièrement décentralisée et qui s'articule autour de plusieurs niveaux. Au niveau le plus bas, un réseau dédié P2P est construit, reliant chacun des nœuds dans le réseau les uns avec les autres.

II.5.2 ALP : Advanced Logistics Project

En 1996, la *Defense Advanced Research Projects Agency* (DARPA), en collaboration avec la *Defense Logistics Agency* (DLA) ont commencé un effort de recherche d'environ 80 millions de dollars appelé le projet de logistique avancée : *Advanced Logistics Projetc* (ALP), visant à développer la prochaine génération de systèmes logistiques (Carrico et Greaves, 2008). Ce projet se base principalement sur les stratégies suivantes :

- *« Gathering »* : qui consiste à récupérer les informations du monde réel.
- *« Supporting »* : qui consiste à recevoir/identifier les tâches ou les changements importants.
- *« Decomposing »* : qui consiste à décomposer un problème global en plusieurs sous-problèmes.
- *« Acting »* : qui consiste à exécuter une action précise en interaction avec les entités du monde réel.
- *« Delegating »* : qui consiste à allouer les sous-problèmes aux ressources disponibles.
- *« Assessing »* : qui consiste à contrôler continuellement l'évolution pour être sûr que tout se passe conformément à ce qui a été prévu .
- *« Reporting »* : qui consiste à préparer des rapports pour le monde extérieur.

Les concepteurs utilisent un moteur de *workflow* dynamique pour construire les modèles, interconnecter les représentations des composantes cognitives et accomplir les tâches.

II.5.3 Ushahidi, logiciel libre pour la collecte d'informations et la cartographie interactive des données

Ushaihidi est décrit par ses créateurs comme un logiciel permettant aux gens, où qu'ils soient dans le monde, de raconter ce qu'ils se passent autour d'eux pendant une situation d'urgence (le mot « ushaihidi » signifie *témoignage* en swahili). Ce système, voulu accessible partout et facile à réaliser, a été développé durant les émeutes post-électorales au Kenya en 2008 pour venir en aide aux victimes des violences. A partir de tout appareil électronique pouvant transmettre des données, le logiciel conçoit une carte centralisant les informations clés lors d'une crise. Les usagers envoient un *sms*, un *mms* ou un courriel et les données apparaissent dessus.

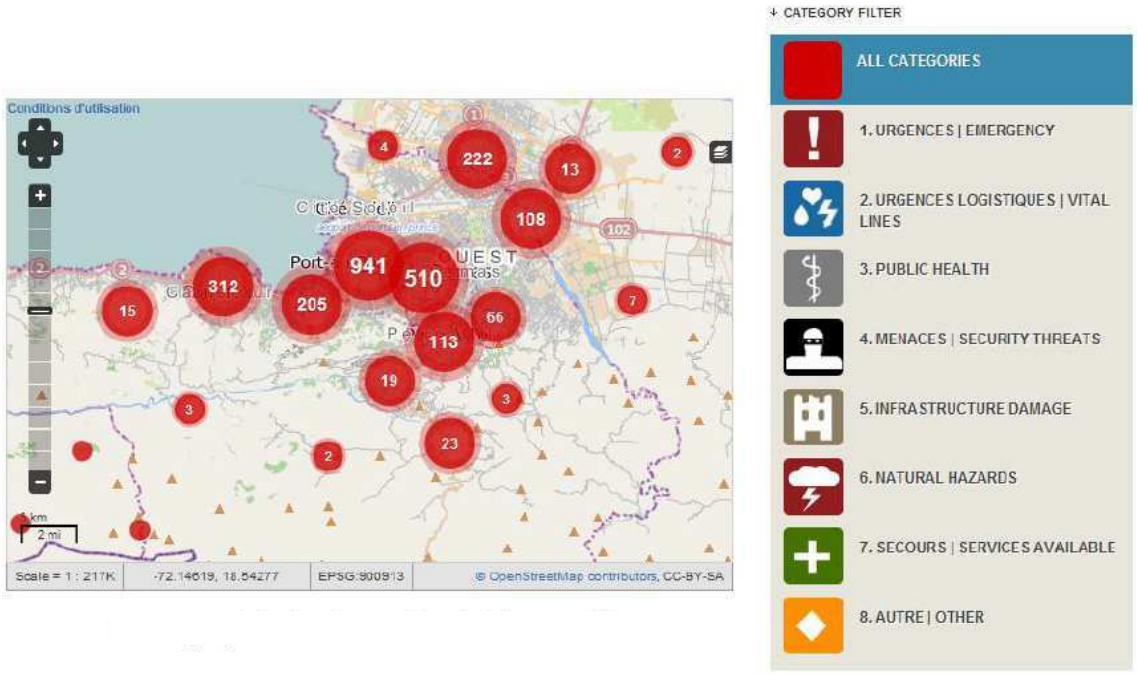

Figure II. 7. Cartographie des demandes d'aide en Haïti

Le système a par exemple été utilisé en Haïti après le séisme de 2010 (http://haiti.ushahidi.com).

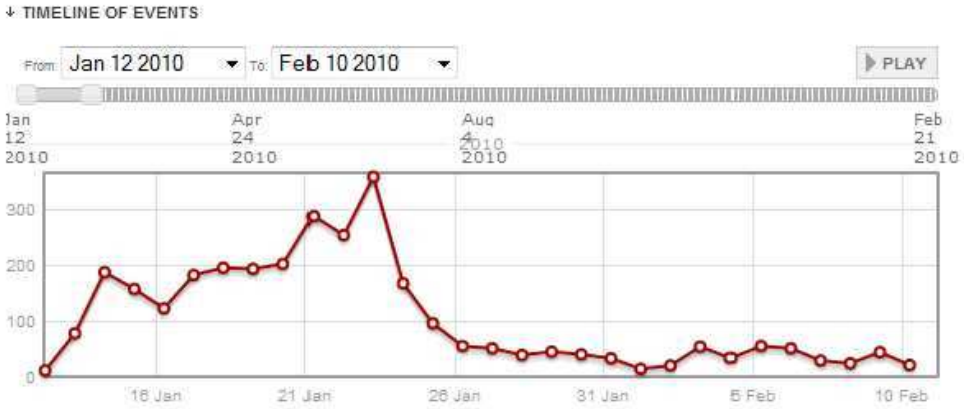

Figure II. 8 Calendrier des événements

II.6 Conclusion

Dans ce chapitre, nous avons présenté le concept d'optimisation, les principales notions qui y sont reliées et une brève revue des systèmes existants dans la littérature et qui ont traité de ce concept. En second lieu, les systèmes multi-agents ont été présentés en mettant en exergue les contributions qu'ils sont susceptibles d'apporter notamment lorsqu'il s'agit de traiter des problèmes d'importante complexité, comme c'est le cas de la problématique de gestion des chaînes logistiques, et plus particulièrement pour la gestion de crise.

Considérant la problématique ainsi énoncée, nous adoptons dans notre approche un croisement mutuel entre ces deux concepts. Dans le but premier de réaliser efficacement des traitements optimisés et dans des délais de réponse raisonnables, nous avons en effet opté pour une alliance des SMA et de l'optimisation dans une architecture spécifique, que nous détaillons dans les chapitres suivants.

Chapitre III : SMA au service de la modélisation avancée d'une chaine logistique

III.1 Introduction

Etant placés dans un environnement instable et fortement évolutif, la mise en place d'une modélisation appropriée qui cadre bien avec le degré de réactivité attendu en cas de crise s'impose. En effet, la forte variabilité des contraintes, données et paramètres de l'environnement s'opposent à la mise en place d'une chaîne logistique de gestion de crise dynamique, optimisée et performante.

Les données du système de gestion de crise sont éparpillées sur un réseau géographique d'une étendue assez large. La distribution de ces données est fortement aléatoire et inconstante. Elle nécessite de ce fait une modélisation qui en suit le mouvement et s'imprime de leurs variabilités afin de représenter en temps réel, l'évolutivité de l'environnement qui les incorpore.

La suite de ce chapitre est organisée comme suit : afin de pouvoir porter un choix sur la mise en place de l'architecture la mieux appropriée, nous proposons dans la première section de comparer les architectures systèmes les plus utilisées dans le domaine du génie logiciel. Quelques notions sur le concept objet et la méthode UML sont présentées dans la section III.3. Ensuite l'architecture du système proposé est détaillée dans la section III.4. Les détails d'implantation du concept multi-agent dans notre approche sont par la suite fournis au niveau des sections suivantes.

III.2 Choix des architectures des systèmes complexes

La spécification de notre problème nous impose plusieurs contraintes. Nous essayons donc au début de ce chapitre de comparer les différentes technologies. Dans un premier temps, nous présentons les technologies les plus utilisées dans le domaine du génie logiciel afin de pouvoir les comparer. Ensuite, nous détaillons les contraintes auxquelles notre système est confronté et nous ferons notre choix d'architecture.

III.2.1 Technologies du Génie Logiciel

Le besoin croissant en systèmes d'information à la fois complexes et performants, a créé une nouvelle demande en « architecture système ». L'apparition de nouveaux concepts structurels et architecturaux en génie logiciel a répondu à cette demande. Ainsi, nous disposons actuellement d'une multitude d'outils et de technologies permettant la création et la gestion

des architectures complexes. Parmi ces concepts, nous présentons les plus répandus, à savoir les composants logiciels, les systèmes multi-agents et les services web. Dans ce qui suit, nous analysons les architectures relatives à ces concepts qui sont : **les architectures à base de composants, d'agents et de services**.

III.2.1.1 Architectures à base de composants

Cette technologie est issue des approches orientées objets. C'est une réponse à la demande de la réutilisation du code dans le développement des applications informatiques. Depuis les années quatre-vingt-dix, cette nouvelle vision des logiciels informatiques ne cesse d'évoluer (Maurer, 2000). Elle consiste à considérer le logiciel informatique comme un ensemble de plusieurs composants (De même qu'un circuit électrique est constitué de composants électroniques assemblés).

Comme c'est aussi le cas pour l'orienté objet, l'un des principaux atouts de la programmation à base de composants est la réutilisation (Barbier et al., 2002). La notion de composant vient compléter les limites de l'approche objet en terme de granularité de réutilisation en proposant une définition plus précise des entrées et sorties de chaque composant et en assurant une meilleure sécurité de la réutilisabilité des composants. En effet, à la différence de « l'objet » qui risque de faire des réutilisations non sécurisées (suite à des appels à des services externes sans spécification explicite), un composant ne peut utiliser que des services bien spécifiés compte tenu de sa conception et des précisions sur ses interfaces (Feki, 2010).

III.2.1.2 Architectures à base d'agents

Comme énoncé précédemment, les systèmes à base d'agents tiennent leurs origines dans le développement de l'intelligence artificielle. En effet, le concept d'agent est une évolution de l'intelligence artificielle distribuée née de la fusion entre les systèmes distribués et l'intelligence artificielle (Chaib-Draa, 1994).

Pour résoudre les problèmes les plus complexes pouvant représenter une limite des capacités de résolution d'une entité (agent) unique, la mise en commun des connaissances de plusieurs agents et leur coopération permet de franchir cette limite et de donner plus d'élan dans les capacités de résolution. Le concept de coopération dépasse ici la simple addition de connaissances et représente une réelle fusion d'un savoir global qui offre une vue plus synthétique sur les problèmes.

III.2.1.3 Architecture orientée services

L'architecture orientée services (ou *Service Oriented Architecture*, SOA) est une nouvelle organisation applicative permettant une interaction entre des composants distants d'une application à travers des services. La notion du service peut être représentée ici par un objet produit par un fournisseur et consommé par un client. La SOA propose un nouveau concept facilitant l'échange de messages, réutilisable et avec une bonne sécurité (utilisation de protocoles standardisés).

L'architecture orientée service est d'après Rouillard (Rouillard, 2007) « une manière d'organiser les applications logicielles isolées à travers une mise en place d'une infrastructure accueillant un ensemble de services interconnectés. Chaque service étant accessible au travers de standards et de protocoles d'échange de message ». Les architectures orientées services représentent une vision basée sur l'offre et la demande de services. Il peut s'agir de tout type de service respectant un/des protocole(s) et une description précise mise à la disposition du client (par exemple les WSDL - pour *Web Services Description Language*).

III.2.2 Comparaison des architectures

Pour analyser au mieux les différentes architectures, nous nous réfèrerons aux recherches effectuées par Briot (Briot, 2009) dans le cadre de son analyse comparative des architectures à base de composants logiciels et les systèmes multi-agents.

Dans son analyse, Briot propose de comparer les différents concepts en se projetant sur un référentiel commun à trois dimensions, où chaque dimension traite un axe important de la programmation et du logiciel. Ces axes sont les suivants: la sélection de l'action, le niveau d'abstraction et la flexibilité du couplage.

III.2.2.1 Sélection de l'action

Elle identifie quand et comment l'entité logicielle active l'action du code correspondant. Globalement, on remarque que « la sélection de l'action » a été repoussée à plus tard dans les compilateurs avec l'évolution des langages de programmation.

En effet, dans les premiers langages (Ex. Fortran), les différentes instructions sont identifiées par leur numéro de ligne. La sélection de l'action est par conséquent exprimée globalement et statiquement. Ensuite sont apparus les langages de programmation structurée ou modulaire, (Ex. Pascal), qui innovent par une encapsulation du code dans des procédures. Cette modularité implique un appel par un nom symbolique et non pas par un numéro de ligne. L'invocation n'est donc plus globale et devient externe. Cependant, l'identification de la sélection de l'action demeure statique.

La grande innovation des langages orientés objet dans ce domaine est l'apparition de la liaison tardive (*late binding*) qui consiste à ce que la méthode à appeler soit identifiée selon son appartenance à un objet. Ainsi la sélection de l'action est repoussée jusqu'à l'exécution et n'est plus identifiée à la compilation d'une manière statique, la sélection est donc externe et dynamique.

D'après Briot (Briot, 2009), les composants logiciels n'innovent pas particulièrement en termes de sélection de l'action. La seule différence par rapport à la programmation-objet réside dans le fait qu'un composant n'a pas besoin d'information externe pour son exécution : l'intégralité du code ainsi que la documentation d'un composant se trouvent dans le composant lui-même (Oussalah, 2005). Par opposition, pour un objet, une partie de son code se trouve dans sa classe mère. Les agents apportent une grande évolution à la sélection de l'action. En effet, contrairement aux procédures et méthodes qui sont invoquées à un lieu particulier du code, la sélection de l'action dans le concept agent (comportement) dépend aussi du contexte, de l'état et des connaissances de ce dernier. Cette nouvelle autonomie à la sélection de l'action rend l'agent proactif en plus de son aspect réactif. Ainsi, la sélection de l'action devient interne et dynamique.

III.2.2.2 Flexibilité du couplage

Elle est définie par la capacité à mettre en relation plusieurs entités logicielles. On distingue deux types de couplage : le couplage structurel et le couplage de communication. Nous nous concentrons dans le paragraphe suivant sur le couplage structurel.

Le couplage structurel est mis en place dans les langages de programmation à travers les références. En effet, un objet X référence un deuxième Y en mettant simplement la référence de Y dans une variable locale à X qu'on notera V_x. Donc pour changer la référence de X vers un autre artéfact Z par exemple, il suffit de changer la valeur de la variable V_x en lui affectant Z. Un problème majeur intervient si on souhaite que X référence en même temps plusieurs artéfacts (Z_1...Z_n). Dans ce cas, on est obligé de passer par des collections informatiques telles que les listes, les files, les piles, etc. En plus, il faudra changer les instructions communiquant avec la référence vu que ce n'est plus un objet, mais une collection d'objets qui est traitée. Il y aura donc un ajout d'itérateur de collection et ceci nous oblige à ré-implémenter le code de l'objet X.

La nouveauté des portes/interfaces de sorties dans l'architecture des composants permet de palier au problème décrit précédemment. Un composant ayant un ensemble de bornes d'entrée et un ensemble de bornes de sorties, facilite la manipulation du couplage en l'externalisant.

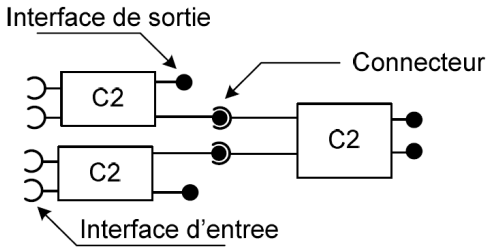

Figure III. 1 Connexion de plusieurs composants

L'architecture SOA (*Service Oriented Architecture*) et en particulier les web services rendent les couplages des différentes entités applicatives dynamiques à travers la consultation de l'annuaire et les choix du service.

Les systèmes multi-agents intègrent aussi cet aspect dynamique de sélection du service à exécuter vu qu'ils implémentent entre autres des agents annuaires. Ce dernier type d'agent existant dans l'architecture FIPA[1] et permet aux autres agents d'un système de retrouver la liste d'agents fournisseurs d'un service en particulier. C'est un annuaire sous forme d'agent qui facilite la mise en relation avec un agent proposant un service recherché.

De plus, les systèmes multi-agents proposent un couplage sémantique basé sur une organisation sociale et des connaissances. Ceci fait évoluer les langages de programmation et les logiciels vers un niveau plus élevé en connaissance avec l'utilisation du concept d'organisation. En outre, la séparation de la description de l'architecture du système d'une part et de l'implémentation d'autre part, représente un grand pas dans l'évolution de la programmation. Ceci est confirmé dans les architectures à base de composants. Les composants étant des entités indépendantes avec des descriptions précises, l'architecture du système assimilé à une carte électronique constituée de composants interconnectés, est totalement indépendante de son implémentation. Cette avancée technologique est encore plus développée dans les architectures service grâce à la découverte, le choix automatique et l'invocation des services. D'autant plus chez les architectures multi-agents avec leur fameuse collaboration, organisation, négociation et coordination.

III.2.2.3 Niveau d'abstraction

Le niveau d'abstraction est défini par le niveau d'expression des concepts offerts aux concepteurs et aux programmeurs. On remarque aisément la grande corrélation entre les langages de programmation et le niveau d'abstraction. En effet, dans les premiers langages de programmation (très proches de la machine), on se basait sur les instructions et les numéros de lignes. Ensuite apparaissent les langages modulaires (procédure, fonctions) avec un degré

[1] FIPA : The Foundation for Intelligent Physical Agents

plus haut d'abstraction, qui continue à augmenter avec l'apparition des structures de données et surtout la programmation-objet qui nous fait évoluer des données vers des concepts via le principe des objets informatiques représentant des objets physiques (Perrot, 1998).

Les composants n'innovent pas sur cet axe contrairement aux agents qui permettent d'étendre cette évolution vers une vision à base de connaissance. En effet, les agents cognitifs par exemple, apportent la notion d'état mental qui est une modélisation symbolique de concepts cognitifs qui permettent une coordination et une organisation entre les agents.

III.2.2.4 Synthèse

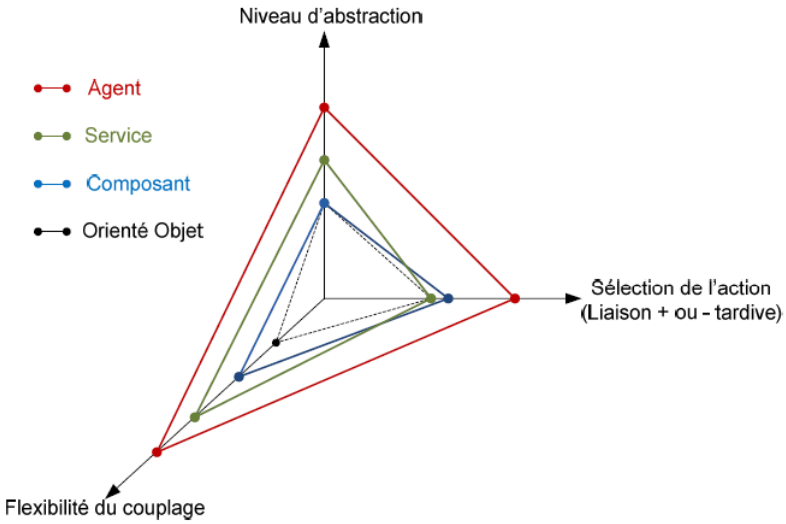

Figure III. 2 Comparaison des technologies

L'architecture composants est très proche de l'architecture orientée objet, elle présente une légère évolution au niveau de la « sélection de l'action » et de la « Flexibilité du couplage ». Cependant, son « niveau d'abstraction demeure limité ». L'architecture orientée service présente un niveau plus élevé d'abstraction et une meilleure flexibilité de couplage, mais pas d'innovation dans la « Sélection de l'action ». Finalement, l'architecture multi-agent est la plus évoluée sur les trois axes (Figure III.2). Sa faiblesse existe peut-être ailleurs, au niveau de la mise en place ou des performances.

III.2.3 Contraintes

Afin d'offrir une aide à la décision fiable en cas de crise, notre système doit pouvoir accéder en temps réel aux données qui caractérisent la situation logistique réelle sur le terrain. Il doit donc accéder aux différents systèmes d'information et capteurs existants et s'adapter automatiquement à leurs offres de services. Ensuite il faut pouvoir appliquer des algorithmes

d’optimisation pour le positionnement des zones et l’acheminement des ressources, tout en gérant l’intégration à chaud de nouveaux modules.
Les contraintes à respecter sont les suivantes :

- Communication inter-SI : L’architecture de notre système doit pourvoir faciliter l’échange de messages entre le système lui-même et son environnement. En effet, la base de son fonctionnement réside dans la récupération automatique de données à partir du terrain et à les transmettre à ses acteurs. Cette transmission nécessite une forme particulière d’échange de messages entre les entités du système et doit respecter certains standards. D’autre part, la récupération de données à partir d’autres systèmes d’informations doit présenter une certaine forme d’intelligence.

- L'évolutivité des systèmes**:** Ayant une structure dynamique, les acteurs doivent pouvoir s’inscrire ou quitter la chaîne logistique à tout moment. Dans tous les cas, notre système ne doit pas être affecté dans sa globalité. En effet, il doit permettre une inscription et une désinscription automatique des acteurs logistiques sans avoir à modifier le code source ou même à redémarrer le système. Ceci est fait dans l’optique d’avoir une meilleure fiabilité en cas de perturbation.

- La distribution et l'incomplétude de l'information: Un acteur logistique va devoir accéder à des informations incomplètes et partagées le long des frontières du système.

- Les capacités décisionnelles: Apprentissage et raisonnement sont nécessaires à la prise de décision individuelle et collective pour les acteurs. Ils doivent être dotés de capacités de raisonnement, pour pouvoir acquérir ou modifier les connaissances et ceci en interaction avec l’environnement.

- Calcul complexe : La résolution des problématiques de positionnement optimisé et de l’acheminement des ressources suite à des perturbations nécessite l’application d’algorithmes complexes et une adaptation au calcul et à la résolution de problèmes.

III.2.4 Choix stratégiques

Sans surprise, nous remarquons que les trois technologies répondent à la première contrainte à savoir la communication entre SI. Mais leur forme d’échange de messages diffère.
Comme précisé précédemment, nous sommes confrontés à deux types de transfert de messages :
Un transfert intra-système : qui consiste dans les transferts de messages entre les différentes entités du système.

Un transfert inter-système : qui consiste dans les transferts de messages entre notre système et les autres systèmes d'informations.

Sachant que l'architecture agent est la plus développée en matière de transfert de message puisqu'elle n'a pas besoin de connaître l'interface de son interlocuteur contrairement à l'architecture composant et service, nous estimons que cette technologie est la plus adaptée à notre problématique. Nous sommes donc tentés de l'utiliser dans les deux formes d'échange de messages de notre système. Cependant, pour les SI qui n'utilisent aucune des architectures citées, il serait peut-être plus simple de mettre en place une offre de service (WS) permettant l'accès à leur calculateur que de concevoir une architecture multi-agent offrant le même service.

En conclusion, nous préconisons donc l'utilisation de l'architecture multi-agent en interne du système pour l'échange des messages intra-système, et des WS pour l'échange de messages avec les SI (inter-systèmes).

Concernant la deuxième contrainte, l'architecture multi-agent est encore la plus adaptée à ce type de problème. En effet, grâce à la souplesse et l'intelligence de ses agents, l'évolutivité du système est facilitée. L'architecture à base de services, de son coté, apporte un dynamisme qui se traduit par la découverte et l'invocation des services qui peut être très utile pour une adaptation aux SI, mais ceci reste très rigide (pas d'adaptation automatique) et nécessite un effort de développement supplémentaire en plus de la difficulté d'intégration à chaud.

Notre système doit permettre l'intégration de nouveaux acteurs ou de les désinscrire sans arrêter le fonctionnement global du système. Dans le cadre de l'architecture à base de composants, les nouveaux composants doivent être conçus en amont et ensuite implémentés avec le système ce qui rend l'intégration de composants à chaud difficile. Les agents quant à eux peuvent être implémentés indépendamment les uns des autres et il est, par conséquent, très simple de démarrer ou d'arrêter un agent lorsque le système tourne sans aucun impact sur le fonctionnement global. Enfin, notre problématique nous impose l'exécution d'un processus d'optimisation global, et donc des méthodes de résolution, dans un environnement distribué.

Dans ce genre de contexte, différentes entités doivent exécuter des instructions indépendamment des autres et communiquer entre elles par la suite pour regrouper les différents résultats dans une réponse globale. Les trois architectures permettent cette approche avec un léger désavantage pour l'approche service qui par concept n'est pas destinée au calcul algorithmique et d'une manière générale n'est pas l'hébergeur idéal pour une masse de calcul. Les deux autres technologies, quant à elles, sont très utilisées dans la recherche algorithmique distribuée.

III.2.5 Principales méthodes existantes à base d'agents

Dans cette section, nous allons présenter les principales méthodes orientées agent. Chaque méthode sera décrite brièvement puis nous examinerons plusieurs axes de comparaison. Ces méthodes, pour la plupart, sont disponibles en ligne, les outils associés étant le plus souvent libres.

III.2.5.1 ADELFE

ADELFE (*atelier de développement de logiciels à fonctionnalité émergente*) est dédiée à des systèmes multi-agents très spécifiques : auto-organisateurs par coopération. Les concepts manipulés par ADELFE sont directement issus des AMAS (*adaptive multi-agent systems*) (Georgé et al., 2003). En effet, l'adaptation du système est obtenue par *interactions* coopératives entre les agents autonomes ayant des buts locaux. Ces derniers sont intégrés dans le module de compétence des agents. De même, les croyances peuvent être vues comme des représentations, qui sont forcément locales et relatives. L'autonomie d'un agent est assurée par l'encapsulation des compétences et des représentations dans les modules de l'agent, ainsi que par l'accès privé aux décisions de l'agent qui garantit la proactivité. La réactivité des agents est obtenue par l'utilisation de machines à états finis pour transcrire les comportements dynamiques et sociaux. Les interactions sont en effet prises en compte dans le module d'interaction – *via* les actions et les perceptions.

ADELFE utilise UML (*unified modeling language*) et AUML (*agent UML*). Les agents sont vus comme des agents stéréotypés possédant des attributs et opérations stéréotypés en fonction des modules. La prise en compte de la complexité ne passe que par la structuration en paquetages, modèles et sous-systèmes, mais n'est pas aussi efficace que dans Gaia ou MaSE.

III.2.5.2 Gaia

Gaia se veut être générale, applicable à n'importe quel domaine et compréhensible par la distinction entre macro-niveau et micro-niveau. Les agents modélisés, pouvant être hétérogènes, sont des systèmes computationnels à gros grain, qui vont essayer de maximiser une mesure de qualité globale. Toutefois, Gaia ne prend pas en compte les systèmes admettant de réels conflits. L'organisation (d'une centaine d'agents environ) est clairement statique dans le temps, de même que les services offerts par les agents.

Gaia manipule six modèles d'analyse et de conception différents :

– le *modèle de rôle*, qui identifie les différents rôles devant être joués par les agents du système ;

– le *modèle d'interaction* qui définit les protocoles de communication entre agents ;
– le *modèle d'agent* qui attribue les rôles aux différents agents du système ;
– le *modèle de service* qui définit, comme son nom l'indique, les différents *services* offerts par le système, et les agents tributaires ;
– le *modèle d'organisation* qui définit la structure de l'*organisation* grâce à des graphes orientés représentant les voies de communication entre agents ;
– le *modèle environnemental* qui décrit les différentes ressources accessibles caractérisées par les types d'actions que les agents peuvent entreprendre pour les modifier.

III.2.5.3 INGENIAS

INGENIAS (Gomez, 2002) est la continuité du projet MESSAGE (*methodology for engineering systems of software agents*) qui étend la notation UML et s'intègre dans l'USDP (*unified software development process*).
Dans INGENIAS, les concepts agents sont manipulés dans des vues (ou modèles) :
– la *vue de l'organisation* montre les entités concrètes dans le système et son environnement, et la relation de haut niveau comme l'agrégation, le pouvoir ou les accointances (qui implique la définition d'interactions) ;
– la *vue des buts*/*tâches* montre les buts, les tâches, les situations et les relations entre eux. Les dépendances temporelles peuvent être définies, ainsi que les dépendances d'ordre compositionnel ;
– la *vue des agents*/*rôles* associe les rôles aux agents, en spécifiant les événements déclenchant l'attribution de rôles aux agents et les ressources exploitées ;
– la *vue des interactions* définit, pour chaque interaction agent/rôle, l'initiateur, les collaborateurs, les « motivateurs », les informations échangées, les événements déclenchant les interactions, et les autres effets possibles comme l'attribution de nouveaux rôles par exemple ;
– la *vue du domaine* est une vue classique en analyse orientée objet qui décrit le domaine d'application du système en termes d'entités et de relations entre elles.

III.2.5.4 MaSE

MaSe, pour *Multiagent Software Engineering*, propose une approche complète pour le développement des systèmes multi-agents. Cette approche couvre toutes les étapes de l'analyse jusqu'au déploiement. Dans cette méthodologie, et comme dans Gaia, l'autonomie des agents est assurée par l'encapsulation des fonctionnalités dans les rôles. Elle utilise des techniques de modélisation directement inspirées de la conception objet, à savoir de l'OMT

(*Object Modeling Technique*) et UML (*Unified Modeling Language*). Deux étapes principales composent cette méthodologie :

- L'analyse qui consiste à identifier les rôles et les cas d'utilisations, créer les diagrammes de séquences et transformer les buts en ensembles de rôles.
- La conception qui a pour but d'assigner les rôles à des classes d'agents spécifiques, de définir les classes internes des agents et la structure finale du système.

Cependant, les architectures développées avec MaSE sont statiques, fermées et ne permettant pas de définir des sociétés à grand nombre d'agents. O-MaSE (*Organization – based Multiagent Software Engineering*), une extension à la méthodologie MaSE, la complète par la dimension d'organisation, et définit un méta-modèle permettant aux agents d'adapter leur organisation à l'exécution (DeLoach, 2005).

III.2.5.5 PASSI

PASSI (*Process for Agent Societies Specification and Implementation*) est une méthode intégrant des modèles de conception et concepts inspirés de l'ingénierie orientée objet (Cossentino, 2001). PASSI réutilise fortement UML et manipule beaucoup de concepts communs aux agents en y rajoutant des notions propres à FIPA. Ces notions sont encapsulées dans cinq modèles : le modèle des besoins, le modèle de sociétés d'agents, le modèle d'implémentation d'agent, le modèle de codage et le modèle de déploiement. C'est une méthode pouvant s'appliquer à n'importe quel domaine, mais plutôt orientée agents mobiles.

III.2.5.6 Prometheus

Prometheus est le fruit d'une expérience de programmation avec la plateforme JACK, développée par Padgham (Padgham, 2003). Cette méthode reprend les mêmes concepts que dans Gaia ou MaSE. Par contre, bien que la notion de société soit avancée, les notions de rôles sont inexistantes et leurs gestion non abordée. Un point positif par rapport aux autres méthodes est la notion d'incident pouvant survenir en cours de fonctionnement.

Le processus de Prometheus se décompose en trois étapes :

- La spécification du système : identification des buts et sous-buts du système en utilisant des scénarios de cas d'utilisation, définition des données de travail, etc.
- La conception architecturale : spécification des interactions, données et perceptions des agents à travers un diagramme de vue et des diagrammes de séquences, etc.
- La conception détaillée : spécification de la vue interne des agents en précisant les composantes internes, capacités, plans, croyances et données, etc.

Prometheus gère une grande partie du cycle de développement, contrairement aux phases de test, validation et déploiement qui sont presque absentes. C'est une méthode qui convient plus à des applications orientées utilisateurs et moins à la résolution de problèmes ou à la simulation.

III.2.5.7 Comparaison des différentes méthodes

Bernon propose une synthèse de la comparaison des méthodes présentées dans le tableau1.

	ADELFE	*Gaia*	*MaSe*	*INGENIAS*	*PASSI*	*Prometheus*
Adaptation	+ +	±	+	±	±	-
Autonomie	+	+	+	+	+	+
Buts	+	±	+ +	+ +	±	+ +
Croyances	+	±	+	-	+	+ +
Désirs	-	±	+ +	+	+ +	+ +
Interaction/Message	+ +	+	+ +	+ +	+ +	+ +
Rôle	+	+ +	+ +	+ +	+ +	-
Modularité	+ +	+ +	±	+ +	+ +	+

Tableau III. 1 Synthèse de la comparaison des différentes méthodes

Légende : (++) pour les propriétés pleinement prises en charge ; (+) pour les propriétés prises en charge de manière indirecte ; (±) pour les propriétés potentiellement prises en charge ; (-) pour les propriétés non prises en charge.

Nous pouvons constater que les interactions ou les notions d'agents sont communément abordées. L'autonomie reste « à une moyenne de + » seulement. Globalement, et suivant une lecture selon l'axe verticale, un premier constat est qu'aucune méthode n'est parfaite dans ce sens. Chaque méthode possède des spécificités en termes d'architecture, de formalisme ou de modèles. INGENIAS est assez générale et convient à tout type de contexte et propose un processus et des notations riches et expressives, utilisant largement les notions d'UML.

Ainsi placés dans un contexte d'intelligence artificielle distribuée, nous verrons dans ce qui suit la manière dont les fonctionnalités du système proposé ont été dispatchées sur les différentes entités impliquées. Pour cela, une modélisation basée sur les agents est préalablement spécifiée.

III.3 OBAC : Une architecture proposée à base d'agents communicants

Dans nos travaux nous nous intéressons tout d'abord à la technologie UML qui est un langage de modélisation favorisant la conception et la programmation orientées objet aussi bien que la modélisation agent ; passage obligatoire pour la définition d'une architecture logicielle appropriée. Dans ce qui suit, nous définissons les notions de base relatives au concept « objet » avant d'introduire l'architecture proposée pour la modélisation de notre système et par la

suite détailler l'implantation du concept multi-agent. Les comportements des différentes entités ainsi définies, évoluant dans l'environnement propre à la gestion de crise, sont présentés à l'aide de la modélisation ici adoptée.

III.3.1 Notions élémentaires

III.3.1.1 Objet

Un objet est toute entité ayant un sens dans une application quelconque du monde réel. Il s'agit d'un conteneur symbolique qui peut représenter une entité physique (e.g. un individu, un périphérique d'ordinateur …), un concept (e.g. un processus, un procédé d'analyse de données, …) et toute *« chose »* visible et tangible (un matériel, une substance, un produit…) qui possède sa propre existence (Shlaer & Mellor, 1988). Il incorpore des informations et des mécanismes en rapport avec l'entité du monde réel manipulée. Toute la technologie objet, la modélisation et programmation orientées objet tournent autour de ce concept central.

III.3.1.2 Modélisation Objet

La modélisation objet consiste à représenter selon un modèle bien défini en informatique les éléments de l'environnement réel du problème représenté par le système (Wirfs-Brock et al., 1990). Cette modélisation est, dans le cas général, indépendante des outils de programmation mis à profit pour le développement logiciel du système en question. La conception objet revient donc à déterminer les objets présents (i.e. représentant les éléments évoluant dans l'environnement) et isoler leurs données et les fonctions qui les utilisent et en créer des prototypes ou classes pour les représenter.

De multiples langages ont ainsi émergé pour définir les modèles et patrons de conception pouvant être utilisés pour modéliser un problème quelconque. Plus de 50 méthodes objets ont été proposées par différents analystes entre 1970 et 1994. Cependant, trois seulement se sont réellement distinguées. Il s'agit des trois méthodes définies dans la suite : OOSE, Booch et OMT qui ont donné lieu par la suite au Langage de Modélisation Unifié (*UML : Unified Modeling Language*).

III.3.1.3 Classe

Une classe est un modèle (*template*), une représentation abstraite d'un ensemble d'objets ayant des attributs et méthodes communs. Un objet est dit instance d'une classe. Il hérite des caractéristiques (i.e. attributs) et comportements (i.e. méthodes) d'un prototype bien déterminé. Une classe représente ainsi toutes les « *choses »* ayant les mêmes caractéristiques et suivant les mêmes règles. Elle se réfère donc à un ensemble d'objets ayant :

- des propriétés similaires
- des comportements communs
- et les mêmes relations

III.3.2 UML : Langage de Modélisation Unifié

Unified Modeling Language ou langage de modélisation unifié permet de modéliser un problème de manière standard. Il s'agit d'un langage mettant en place un bon nombre de diagrammes pour la modélisation graphique de l'ensemble des données et traitements intégrés au niveau du système à concevoir. Ce langage a été proposé pour réunir, unifier et fusionner trois autres méthodes de modélisation objet, à savoir OMT, Booch et OOSE.

Normalisé par l'OMG en 1997, UML définit formellement l'approche objet tout en lui donnant une dimension méthodologique en se basant sur une multitude de vues du système illustrées à travers la Figure suivante.

Figure III. 3 Les différentes vues d'UML

Les vues décrivent l'ensemble du système d'un point de vue organisationnel, architectural, dynamique, temporel, logique, géographique, etc. Chacune comporte un ou plusieurs diagrammes qui sont au nombre de 13 dans le langage UML dans sa version 2.3. Chaque diagramme est classé sous une catégorie bien particulière, nous en distinguons 3 en tout :

1. Diagrammes structurels ou statiques : réunissent au total 6 diagrammes : de classes, de composants, d'objets, de structure composite, de déploiement et de paquetage.

2. Diagrammes comportementaux : regroupent les diagrammes de comportement, d'activités et d'états-transitions.

3. Diagrammes d'interaction ou dynamiques : modélisent les échanges inter-objets et se réfèrent aux diagrammes de séquences, de communication, de temps ou encore le diagramme global d'interaction.

UML n'étant pas une méthode, le choix des diagrammes revient à l'appréciation de chaque concepteur. Nous nous intéressons particulièrement au diagramme de classes placé sous le signe de la structure statique. Ce type de diagramme sera mis à profit afin de définir l'architecture logicielle de notre système.

Aussi un deuxième type de diagrammes relatif aux diagrammes comportementaux est utilisé pour la modélisation du fonctionnement des différents agents inclus dans l'architecture en question. Afin de modéliser le comportement (i.e. rôle) de chacun de ces agents, nous avons opté pour les diagrammes d'activité pour en faire ressortir les différents reliefs. Un troisième type de diagramme UML correspondant aux diagrammes de séquences classé sous la catégorie des diagrammes d'interaction sera mis à profit pour définir les échanges et communications inter-agents afin d'en modéliser les coalitions ayant lieu d'être.

III.3.3 De la modélisation Objet vers la modélisation Agent

Une conception basée sur les agents peut être vue comme un style architectural qui dérive de la modélisation objet. En effet, l'architecture agent est le produit du passage du composant objectif vers le composant projectif. Un objet étant défini comme étant un composant passif puisqu'il est pourvu d'un ensemble d'attributs et de méthodes à travers lesquels il peut être manipulé. Il offre ainsi un ensemble de services et utilise d'autres objets afin de réaliser ses propres fonctionnalités.

De manière quelque peu similaire, un agent utilise d'une façon intelligente les autres agents pour réaliser ses objectifs. Parallèlement à l'échange de messages entre objets pour établir les interactions possibles pouvant les relier, les agents communiquent de par l'établissement d'un support pour dialoguer grâce à un protocole de communication facilitant ainsi l'envoi et la réception de messages, la négociation et l'échange d'information dans un contexte de coalition inter-agent.

III.4 Architecture proposée à base d'agents communicants pour la gestion de crise

Dans le but essentiel de contrecarrer les problèmes de complexité exponentielle, nous nous proposons de mettre en place un système de résolution qui se base sur la décomposition du processus global en un ensemble de tâches moins complexes s'exécutant en parallèle. Cette idée de base se traduit par la mise en place d'une architecture logicielle distribuée où le problème initial est décomposé en plusieurs sous-problèmes auxquels sont associés des processus plus ou moins indépendants. Dans un contexte de *« multi-threading »*, ces

processus exécutent en parallèle et de manière autonome une ou plusieurs requêtes ou sous-requêtes favorisant ainsi le traitement instantané et parallèle des demandes de livraisons des zones sinistrées. Le concept multi-agent est favorable à la mise en place d'une architecture dynamique distribuée. Ainsi, dans le but d'établir un traitement optimisé efficace et léger en termes de complexité, nous nous proposons de mettre en place un système multi-agent. Ce système baptisé OBAC pour Optimisation à Base d'Agents Communicants est pourvu d'une multitude de fonctionnalités impliquées dans le processus d'optimisation du fonctionnement la chaîne logistique et de la réponse face à la crise. Parmi ces fonctionnalités, nous retrouvons entre autre le positionnement des zones logistiques, le traitement parallèle des requêtes de livraisons de ressources, et l'estimation optimisée des besoins.

L'OMA (Organisation Multi-Agent) proposée est dynamique, de type société, elle considère chaque acteur de la CLGC comme un agent autonome, capable d'échanger des informations avec les autres acteurs. Dans notre CL, les acteurs sont nombreux et variés. De multiples modélisations sont envisageables. Cependant, elles font toutes intervenir les différentes zones de la chaîne logistique, et les modélisent à travers un ou plusieurs agents, que l'on appellera « agents gestionnaires » de ces zones. Ainsi, les échanges d'information, les échanges de marchandises entre zones, vont avant tout résulter d'une communication directe ou indirecte entre les agents gestionnaires des zones.

Pour résoudre le problème décrit précédemment, nous proposons un système dynamique et ouvert, basé sur l'interaction de plusieurs types d'agents logiciels qui vont aider au bon déroulement des échanges de flux physiques et informationnels : Les agents Zones, l'agent Transport, l'agent Intégrateur Evaluateur, l'agent Météo, les agents Interfaces GUI et l'agent Veille. Les agents Interfaces jouent le rôle d'interface entre les utilisateurs et le système, les agents zones sont responsables des zones logistiques et s'occupent des ordonnancements des tâches de livraison en local, l'agent Transport gère l'ensemble des vecteurs de transports de la chaîne logistique, l'agent Intégrateur Evaluateur s'occupent de l'évaluation et la composition des plans de livraison finaux et l'agent Météo fournit des informations concernant les conditions météorologiques.

Un autre type d'agent intervient dans le fonctionnement de la CLGC : l'agent NEA (*Need Estimating Agent*) qui a pour rôle d'estimer les consommations aux niveaux des zones sinistrées.

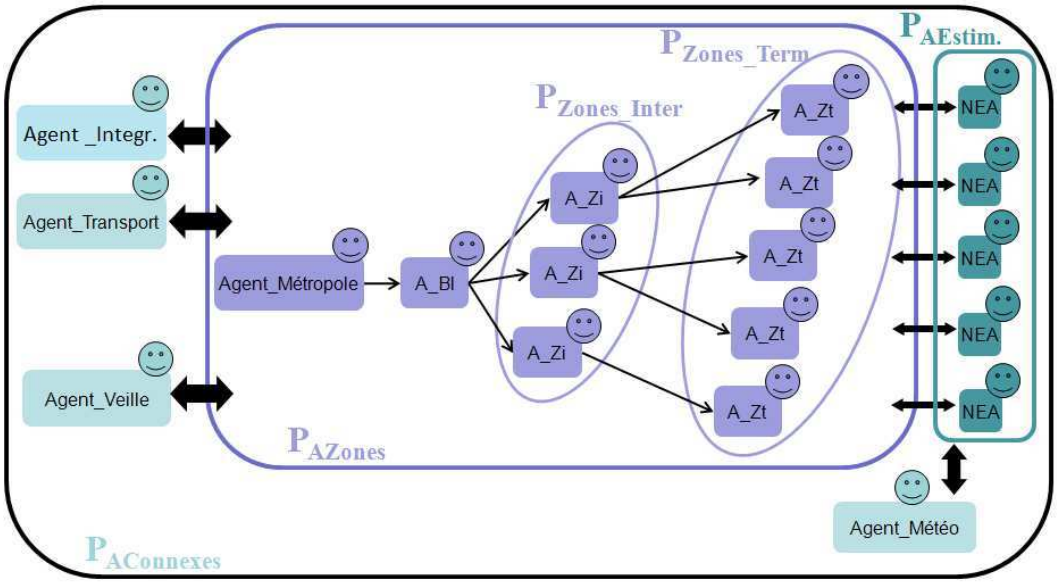

Figure III. 4 Organisation Multi-agent proposée

Comme le montre la Figure III.4, des flux de données sont échangés entre les différents agents aidant à la bonne marche du système dans un contexte de synchronisation évident. Chacune des entités impliquées dans ce processus est pourvue d'un rôle bien défini dans le processus global. L'ensemble de ces entités est défini dans la suite, avec un descriptif détaillé de leurs fonctionnements (Kaddoussi et al., 2012-a).

L'organisation de notre système est illustrée dans la figure III.5.

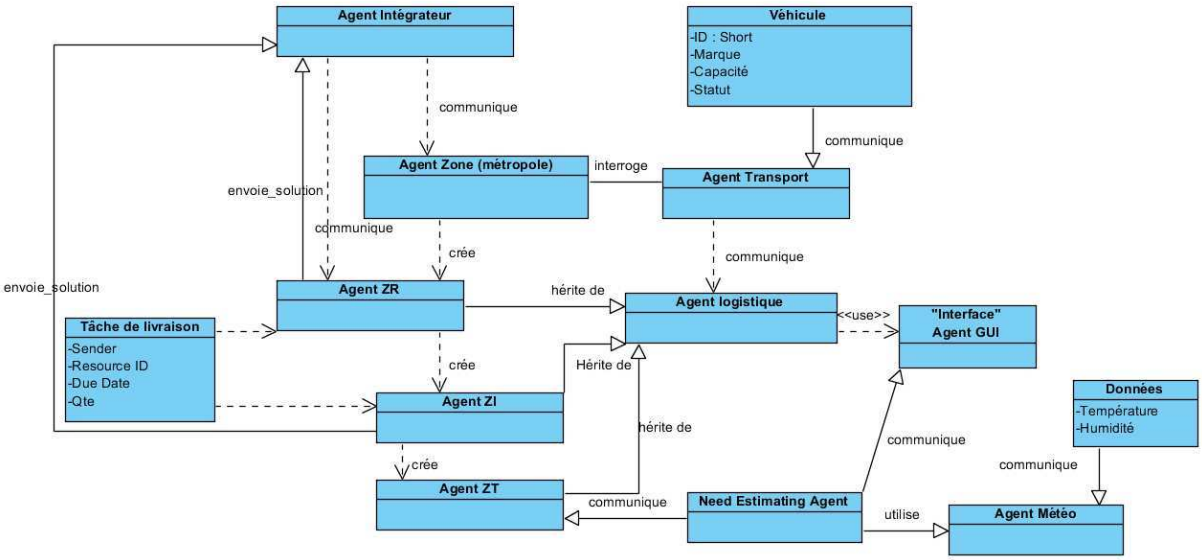

Figure III. 5 Architecture du système

III.4.1 Agent zone

Chaque zone est assimilée à un agent: l'agent-zone. Les flux d'informations, de matériels et humains circulent dans les 2 sens. Un agent-zone peut communiquer avec un agent du même

niveau ou un agent un niveau supérieur/inférieur hiérarchiquement. Chaque agent zone est doté d'un ensemble de comportements qui participent au bon fonctionnement de la chaine logistique de gestion de crise. Parmi ces comportements, nous citons le comportement anticipateur, le comportement négociateur et le comportement ordonnanceur. L'ensemble de ces comportements sera détaillé dans le chapitre suivant. Un agent zone a une durée de vie limitée dans le temps. En effet, il est créé au déclenchement de la crise mais dès qu'il y a un retour à la stabilité et à la situation d'avant crise, cet agent sera détruit. Un agent zone pourra aussi être déplacé si la situation sur le terrain l'impose. La problématique de positionnement optimisé des zones logistiques sera détaillée dans le chapitre 4. Un agent zone est doté des propriétés suivantes :

- Agent organisationnel ;
- Chaque zone de la CL est représentée par un agent zone ;
- Fournit les noms et les adresses des zones supérieures et subordonnées ;
- Fournit les besoins de la zone concernée et zones subordonnées ;
- Fournit le stock de sécurité.
- Fait appel à une méthode de résolution pour ordonnancer les livraisons des marchandises vers ses zones subalternes.

Les agents-zones sont de 4 types. Ils constituent ce que nous appelons une population d'agents zones.

- Agent Métropole : c'est un agent fixe qui gère toutes les ressources de la zone Métropole. Il dispose d'un stock de départ très important et il joue le rôle du fournisseur de toute la CLGC ;
- Agent ZRR : zone de regroupement de ressources. C'est un agent dynamique qui est créé seulement en cas de crise. Au niveau de cet agent, les ressources seront regroupées pour être affectées aux différents moyens de transport possibles. C'est le deuxième maillon de la CLGC. L'opération qui consiste à ramener les ressources de la métropole jusqu'à l'agent ZRR est appelée le pré-acheminement. Il s'agit généralement de zones de regroupement des marchandises en attente d'expatriation en dehors des frontières françaises ;
- Agent ZI : zone intermédiaire. Lorsque la distance entre la zone de regroupement de ressources et la zone sinistrée est importante, une zone logistique intermédiaire est placée et un agent zone intermédiaire est créer pour gérer cette zone.
    - Cet agent est créé par l'agent ZRR ;
    - Il crée autant d'**Agents zones terminales** que de zones sinistrées.

- Agent ZT : zone terminale
    - Il représente la zone sinistrée ;
    - C’est au niveau de l’agent ZT que la consommation de ressources est la plus aléatoire.

Le diagramme d’activités de l’agent Métropole est représenté dans la figure III.6 suivante.

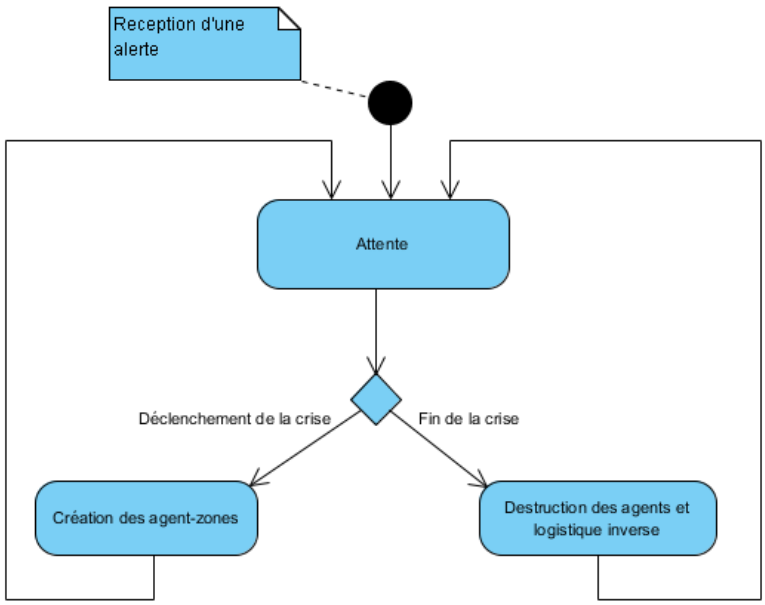

Figure III. 6 Diagramme d’activités de l’agent Métropole

III.4.2 Agent Transport

L’agent Transport s’occupe des transports de marchandises et informe les zones qu’elles viennent de recevoir un colis. Lorsqu’une zone décide d’envoyer des marchandises à une autre zone, l’agent Transport en est informé. Il affecte alors le moyen de transport nécessaire à la livraison et attend un certain nombre de jours, qui correspondent au délai d’acheminement, puis il informe la zone réceptrice qu’un colis est arrivé. Finalement, cet agent s’occupe de l’acheminement des marchandises, et peut être assimilé aux équipes logistiques gérant les camions, bateaux et autres véhicules transportant les ressources voulues.

On remarque que les moyens disponibles pour transporter les marchandises en dehors de la frontière française vers le pays en besoin sont multiples. Selon les destinations et les délais impartis, le choix peut se porter sur la voie routière, maritime ou encore aérienne. Ce dernier choix est en particulier motivé par l’urgence de la livraison, au détriment du coût de l’opération. Ce changement de mode de livraison peut également se faire sur le terrain. Par exemple, nous pouvons imaginer des livraisons par hélicoptère au lieu d’une livraison classique en camion, lorsque l'urgence se fait sentir.

III.4.3 Agent Intégrateur Evaluateur

Cet agent a pour rôle de composer un planning global pour l'acheminement des ressources tout le long de la CLGC, à partir de planning locaux reçus de la part des agents zones.

En effet chaque agent zone responsable de zones subalternes va exécuter un algorithme d'optimisation bien défini, lequel est implanté dans son cœur. Les agents ordonnancent leurs tâches localement dans la zone dont ils sont responsables pour ensuite transférer leurs solutions à l'agent Intégrateur Evaluateur.

Dès lors qu'un ensemble de solutions est généré par le système, l'agent Intégrateur Evaluateur va devoir identifier les paramètres concernés par chaque solution reçue, identifier la nature de la réponse (valide ou pas) et agir en conséquence. L'agent Intégrateur Evaluateur utilise des indicateurs de performance pour vérifier la validité des solutions. En cas de non validité d'une solution locale, l'agent Intégrateur Evaluateur sélectionne l'agent zone concerné et le contacte pour une demande de réajustement.

De plus amples détails sur la manière suivant laquelle est mené le traitement optimisé des tâches de livraison sont donnés avec une description complète des algorithmes alloués à cet effet dans le chapitre IV.

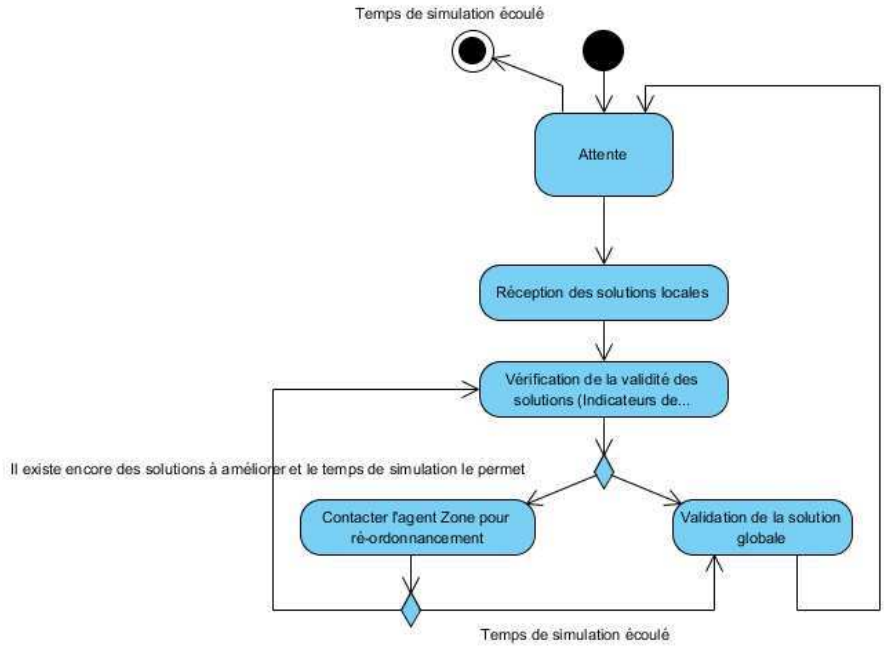

Figure III. 7 Diagramme d'activités de l'agent Intégrateur Evaluateur

III.4.4 Agent Estimateur de Besoins (Need Estimating Agents : NEA)

L'agent estimateur de besoin « *Need Estimating Agent* » (NEA) est un agent holonique ou encore un sous-système multi-agent dont le rôle est de fournir une estimation de la consommation à venir d'un site logistique. Sur une base logistique, il est important de

maintenir ses stocks au-dessus d'un certain seuil critique. Il est possible d'avoir une idée de cette consommation, mais en cas de modifications de certains paramètres, un pic de consommation par exemple dans le cas qui nous préoccupe, cette consommation peut varier énormément. L'objectif du NEA est d'allier l'expertise des personnes habituées à déterminer ces consommations, avec des corrections suivant certains paramètres environnementaux que nous obtenons à partir d'une base de données retraçant les anciennes CLGC et leurs conditions. L'agent NEA scrute donc, de la couche terrain, les informations sur le climat et le nombre de personnes et utilise les modèles d'estimations développés dans la couche mathématique pour simuler et anticiper la variation du phénomène de consommation au niveau des zones logistiques.

Nous décrivons dans ce qui suit le comportement du NEA ainsi que l'évaluation des consommations dans une CLGC et l'organisation de NEA.

III.4.4.1 Comportement du NEA

· L'agent estimateur est appelé par la ZT à laquelle il est rattaché, afin de fournir une évaluation des besoins de la zone ;

· L'agent estimateur contacte l'agent météo, qui lui fournit toutes les informations utiles sur l'humidité et la température dans les 7 jours à venir ;

· A partir de ces informations, il détermine en logique floue un coefficient d'handicap, qui représente la rigueur des conditions climatiques, et qui correspond donc à un besoin en ressources supplémentaires ;

· En tenant compte de la population présente (consommation moyenne des soldats) et du coefficient de handicap pour chaque jour, l'estimateur obtient une première valeur, appelée valeur préconisée non optimisée, du besoin en ressources de la zone ;

· Puis l'agent estimateur effectue une correction de cette valeur, en tenant compte du passé en matière d'estimations, et des différences entre valeur estimée et valeur réelle pour les jours précédents. Nous avons amélioré les modèles d'estimation linéaires existants dans la thèse de (Zoghlami, 2008) en intégrant un estimateur basé sur le modèle ARMAX, modèle qui a prouvé être plus générique et mieux adapté aux situations de crise. Une estimation à base du modèle ARMAX est ainsi réalisée. A la suite de cette correction, l'agent estimateur obtient une deuxième valeur, appelée valeur préconisée optimisée (Kaddoussi et al., 2012-b) ;

· Une fois que la journée est passée, la ZT constate la quantité de marchandises consommées. On informe l'agent estimateur de la valeur réelle. Cela permet à l'estimateur de choisir le mode d'estimation le plus performant, et le plus proche de la réalité, figure III.7.

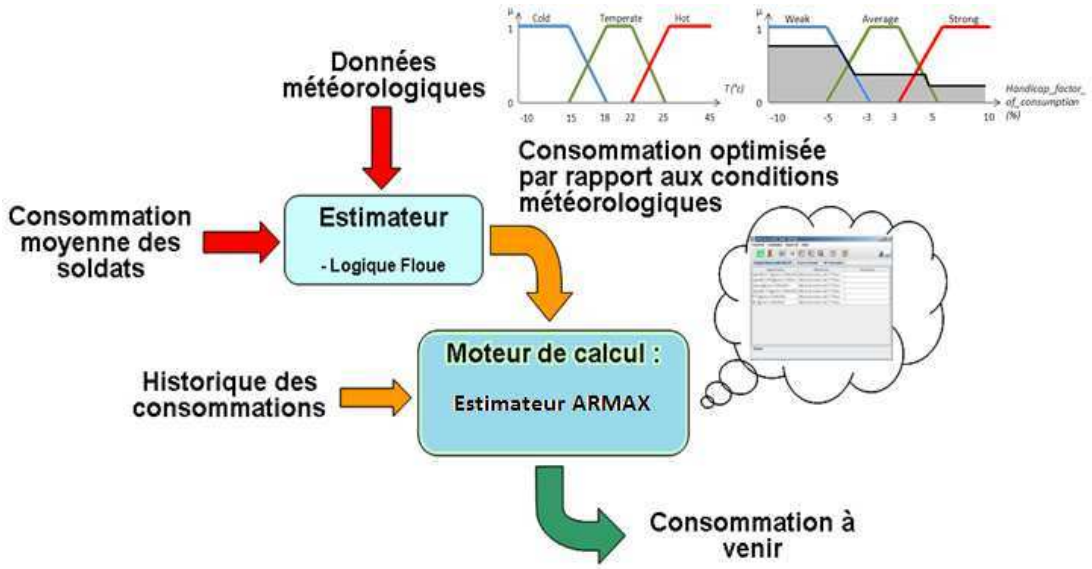

Figure III. 8 Fonctionnement général de l'agent NEA

- Evaluation basée sur la logique floue des consommations dans une CLGC :

A partir d'une consommation moyenne fixe de ressources journalières pour un soldat donné, l'idée est d'anticiper une surconsommation ou sous-consommation en fonction des conditions climatiques à venir.

Il s'agit d'une méthode basée sur la notion de sensation, qui permet de tenir compte de plusieurs paramètres permettant d'aboutir à une estimation. Pour effectuer ces estimations, il est nécessaire de fixer un certain nombre de classes, et de zones permettant de visualiser les degrés de confiance accordés à chacune des sensations. Cette méthode est divisée en trois étapes :

a) La fuzzification :

La fuzzification est la conversion numérique / linguistique des variables d'entrée. Pour notre application, nous proposons de prendre en compte les valeurs de la température et de l'humidité dans la zone considérée pour régler la consommation moyenne de la ressource (exemple : vêtements) et de fournir une estimation précise des besoins futurs dans cette ressource.

Considérons le cas de la température: cette variable d'entrée peut être divisée en trois classes : Froid, Tempéré, Chaud. Prenons l'exemple suivant : en dessous de 15°C, nous considérons qu'il fait froid ; entre 18°C et 22°C, il fait moyen ; et au-dessus de 25°C, il fait chaud. Toutefois, entre 15°C et 18°C, il fait de moins en moins froid et de plus en plus tempéré. De même entre 22°C et 25°C, il fait de plus en plus chaud et de moins en moins tempéré. Cette

étape de paramétrage des classes est réalisée en concertation avec les logisticiens d'EADS. En effet, selon la localisation géographique, la saison, etc., ces valeurs peuvent être changées. Une fois ces valeurs et le nombre de sensations différentes fixés, nous pouvons alors, pour une température T donnée, déterminer son degré d'appartenance dans les trois classes (froid, tempéré et chaud). On obtient ainsi le graphe de sensation, dit fonction d'appartenance, de la température, suivant :

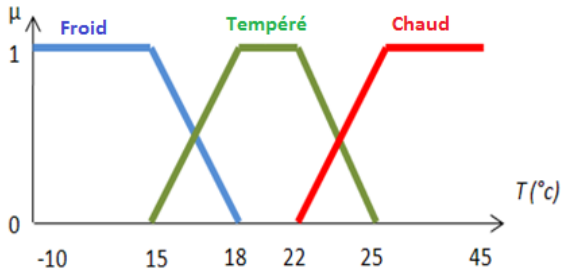

Figure III. 9 Les fonctions d'appartenance aux trois classes Froid, Tempéré et Chaud

Nous procédons de la même manière pour l'humidité.

b) L'inférence floue

L'inférence floue nous permet de développer une décision en utilisant les règles de décision. Ces règles sont basées sur des expériences passées et traduisent l'influence de la météo sur la consommation de vêtements. Les règles de décision sont décrites par des termes linguistiques. La relation entre la température, l'humidité et la variation de la quantité moyenne de vêtements destinés à être consommés sont résumées dans le tableau III.2. Ce tableau se lit de la façon suivante : Si *Température* est « Froid » et *Humidité* est « Sec » Alors *Consommation* est « Forte ». Les règles d'inférence décrivent de ce fait l'influence des conditions climatiques sur la consommation moyenne de vêtements. (Y' aura-t-il une consommation moindre, une surconsommation ou allons-nous maintenir la consommation moyenne?).

	Froid	**Tempéré**	**Chaud**
Sec	Forte	Faible	Faible
Tempéré	Forte	Moyenne	Faible
Humide	Forte	Moyenne	Moyenne

Tableau III. 2 Règles d'inférence

c) La déffuzification

La deffuzzification est la conversion linguistique / numérique des différentes variables. La méthode utilisée dans notre application est la méthode du centre de gravité.

Variable de sortie : Variation de la consommation moyenne.

Nous classons la variation de la consommation moyenne de vêtements en trois classes: Faible, Moyenne et Forte.

- Classe n°1 « Faible » : une sous-consommation par rapport à la moyenne;
- Classe n°2 « Moyenne » : la consommation moyenne est maintenue ;
- Classe n°3 « Forte » : une surconsommation par rapport à la moyenne.

La matrice d'inférence va alors définir les degrés d'appartenance à chacune de ces classes, en fonction de la valeur réelle de la température et de l'humidité.

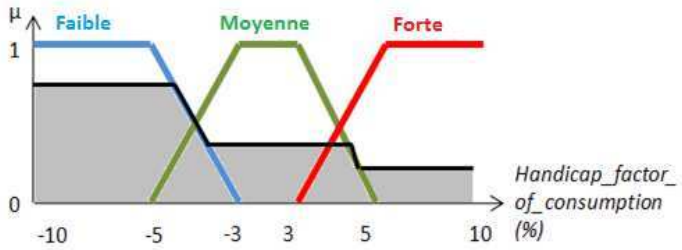

Figure III. 10 Les fonctions d'appartenance aux trois classes Faible, Moyenne et Forte

Le résultat de l'estimation réalisé par la logique floue permet donc d'affiner l'estimation de la consommation par rapport à la consommation moyenne. L'intérêt de la deuxième étape d'estimation est de pouvoir prendre en compte l'historique des consommations pour réajuster la valeur obtenue par cette estimation par rapport aux antécédentes.

- L'identification paramétrique par modèle ARMAX pour l'estimation des besoins

L'intérêt de la deuxième étape de l'estimation est de prendre en compte l'historique des mesures et des consommations afin de redresser la valeur obtenue par la première phase d'estimation floue et obtenir ainsi un modèle mathématique qui représente le phénomène étudié.

Le NEA est un moteur de calcul qui fonctionne par apprentissage, de telle sorte que, plus il calculera d'estimations, plus celles-ci seront précises. Pour ce faire, l'NEA utilise les valeurs obtenues par calcul floue, les valeurs réelles de consommation enregistrées dans le passé et l'erreur obtenue pour améliorer les calculs futurs. Nous proposons d'utiliser l'estimateur ARMAX pour identifier le modèle mathématique qui représente le phénomène mesuré.

Le modèle ARMAX est adapté pour les séries temporelles sans tendance et sans saisonnalité. Il s'agit de la généralisation du modèle ARMA, qui est capable d'incorporer une variable d'entrée externe (X). Le modèle ARMAX s'écrit sous la forme suivante:

$$y^*(n+1) = \boldsymbol{A}^T\, \boldsymbol{y}(n) + \boldsymbol{B}^T\, \boldsymbol{x}(n) + \boldsymbol{C}^T\, \boldsymbol{e}(n)$$

où *y* est un vecteur des variables de sortie observées (ce qui correspond aux quantités réelles de ressources consommées sur le terrain à la fin de la journée de calcul), *x* est un vecteur des variables d'entrée (dans notre cas, *x* est égal au vecteur des variables d'entrée obtenu après la phase de calcul flou), *e* est un vecteur de perturbation (bruit blanc) et *A, B* et *C* sont des matrices polynomiales de la dimension appropriée.

Les étapes du processus d'identification sont comme suit :

a) Identification du modèle: pour construire le modèle ARMAX, il est nécessaire d'identifier ses différents paramètres. Pour ce faire, nous utilisons l'algorithme des moindres carrés récursif (RLS).

b) Model checking: Pour évaluer l'efficacité du modèle mis en œuvre, nous utilisons le critère d'information d'Akaike (AIC). Étant donné un ensemble de modèles candidats pour les données, le modèle préféré est celui avec la valeur minimale AIC. Dans le cas général, l'AIC s'écrit :

$$AIC = -2\log(\varepsilon_l^2) + \frac{2(l+1)}{M}$$

où l est le nombre de paramètres dans le modèle statistique, ε_l^2 est la valeur de la fonction de vraisemblance pour le modèle estimé et M le nombre de mesures. Le critère prend en compte à la fois la pertinence statistique de l'ajustement et le nombre de paramètres qui doivent être estimés pour atteindre le niveau de cet ajustement, en imposant une pénalité pour l'augmentation du nombre de paramètres.

c) Estimations: c'est l'étape de l'extrapolation des données à travers le modèle retenu.

**Algorithme général :*

Initialiser le nombre maximum de variables a_i , b_i , c_i : n_a, n_b, n_c
Initialiser λ, θ(0), Q(0), ε(0) ,... ,ε(n) = 0
Pour i = 0 jusque n_a
Pour j = 0 jusque n_b
Pour k = 0 jusque n_c
Faire pour une étape n
Calculer θ(n) pour le modèle avec la fonction RLS:
$y(n+1) = a_0 y(n) + a_1 y(n-1) + a_2 y(n-2) + \ldots + a_k y(n-i)$
$+ b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) + \ldots + b_{n-l}\, x(n-j)$
$+ c_0 e(n) + c_1 e(n-1) + c_2 e(n-2) + \ldots + c_{n-g}\, e(n-k)$
Retourner θ(n), Q(n), ε(n) ;
Calculer l'AIC correspondant à ce modèle avec la fonction AIC ;
Retourner AIC ;
AIC_min, θ_min, Q_min, ε_min initialisés pour i = 0, j = 0, k = 0 ;
Si AIC < AIC_min
AIC_min = AIC ;
θ_min = θ ;
Q_min = Q ;
ε_min = ε ;
Fin ;
Fin ;
Fin ;
Fin ;
Retourner : AIC_min, θ_min, Q_min, ε_min ; Calcul de y* ;

III.4.4.2 Organisation de l'NEA

Nous avons considéré que le SMA devait être intégralement autonome, et donc n'avoir pour seule communication extérieure que les requêtes qu'on peut lui faire, ainsi que l'envoi des résultats obtenus. De plus, nous avons créé à l'intérieur même du système une structure de mémorisation de l'historique. Cette mémorisation est volatile, c'est-à-dire qu'elle est réinitialisée lorsqu'on arrête le système. Le système s'organise autour de trois organes principaux :

a) Le premier organe est une interface de test qui nous permet de manipuler le NEA et ainsi vérifier son comportement.

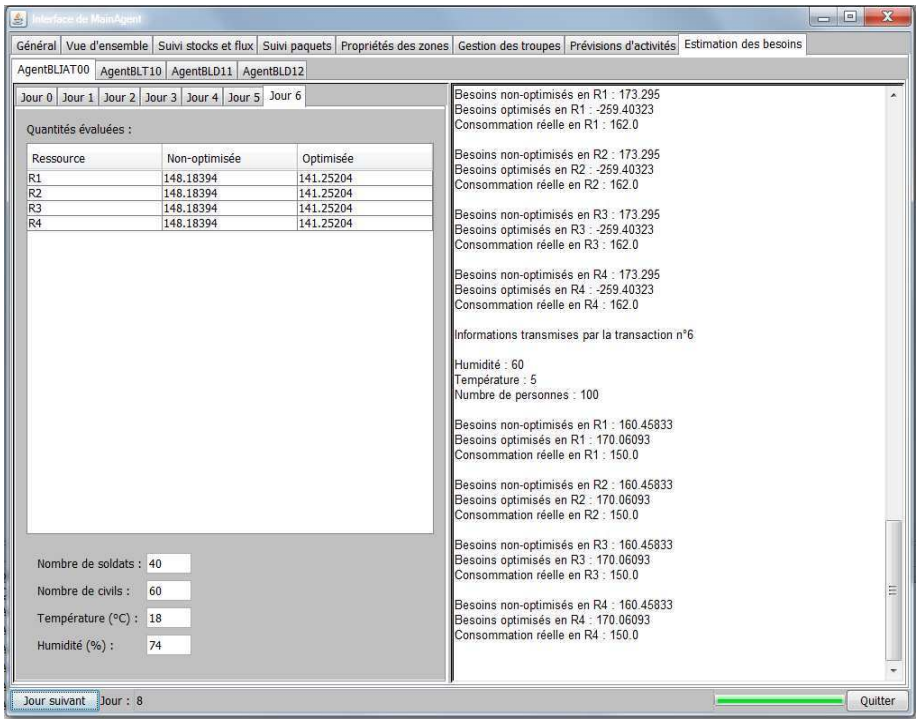

Figure III. 11 Interface de l'agent Test

b) Le second organe est la structure responsable du stockage des données, appelé *Transaction DB*. Son rôle principal est de mémoriser ou de mettre à jour les transactions que nous lui soumettons et de les restituer en cas de demande. Une transaction est un ensemble constitué d'une demande d'estimation, des paramètres nécessaires au calcul de cette estimation, du résultat de cette estimation et d'un feedback réalisé à posteriori sur la qualité de l'estimation.

c) Le troisième agent qui constitue l'NEA est un sous-système multi-agent, constitué de plusieurs agents estimateurs, dont chacun applique un modèle d'estimation différent.

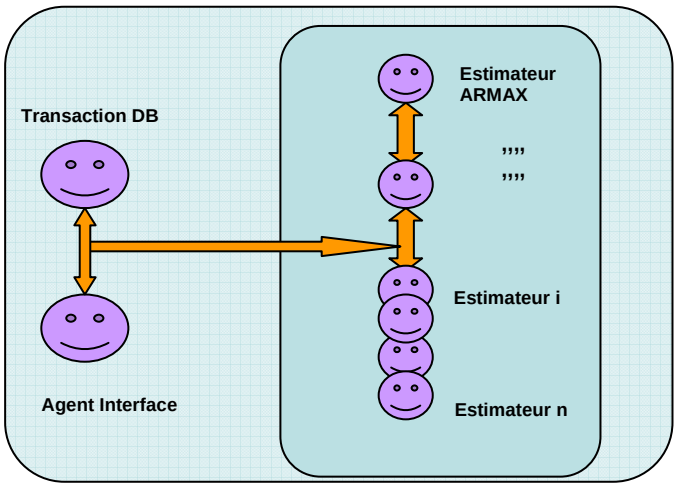

Figure III. 12 Organisation de l'agent NEA

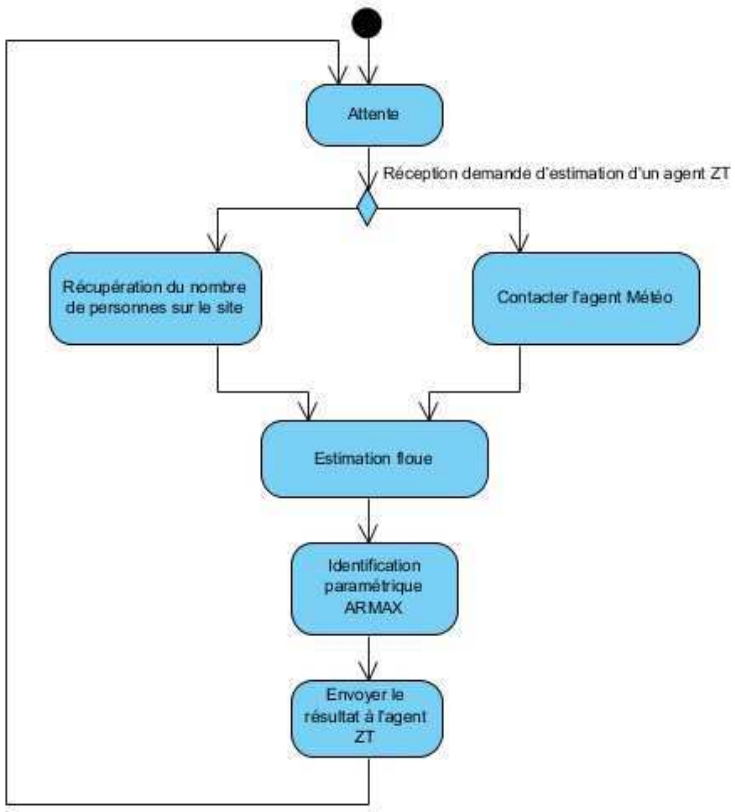

Figure III. 13 Diagramme d'activités de l'agent NEA

III.4.5 Agent Météo

L'agent météo est directement lié aux agents estimateurs ; il donne aux NEA toutes les informations relatives à l'environnement de leur zone ; trois types de données ont été retenus :

- La température ambiante.
- Le degré d'humidité de la zone.
- Le nombre de civils à approvisionner, près de la zone en question.

L'agent météo fournit une estimation de ces données pour le jour-même, ainsi que les 6 prochains jours, ce qui permet ainsi aux agents estimateurs de pouvoir réaliser des estimations précises sur les besoins d'une zone. En effet, on peut constater que ces trois types de données correspondent exactement aux paramètres essentiels dont se servent les agents estimateurs pour effectuer leurs estimations. Il ne leur manque qu'une information : le nombre de militaires présents dans la zone elle-même, qui a également sa propre consommation, et dont l'agent estimateur doit tenir compte. Cette dernière information leur est donnée directement par l'agent zone duquel le NEA dépend.

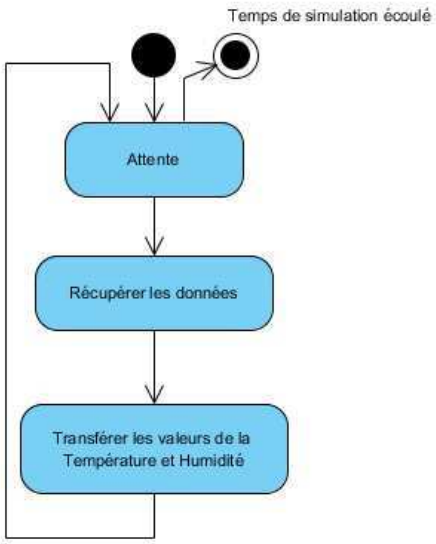

Figure III. 14 Diagramme d'activités de l'agent Météo

III.4.6 Agent GUI

L'agent GUI (Graphic User Interface) est un agent directement rattaché à la fenêtre de contrôle. Il va ainsi dialoguer avec l'ensemble des agents pour leur donner les ordres voulus par l'utilisateur. Il est également chargé de récupérer des informations permettant d'avoir une vision globale de la chaîne à l'écran, et finalement réaliser une étude de risque [Zoghlami et al., 07a]. Dès sa création, l'agent GUI se met en veille attendant tout évènement susceptible de le réactiver et déclencher un processus qui lui est propre.

En réalité, il y a deux agents GUI, pour deux fenêtres et deux types d'interface avec l'utilisateur. La première, correspondant à la chaîne logistique, permet de connaître les informations afférentes à chaque zone ; la seconde correspond à un retour d'informations quant aux estimations de chaque agent estimateur. Les deux agents GUI permettent d'obtenir des informations complètes sur la localisation des marchandises à un jour J, et sur l'état de la chaîne logistique dans son ensemble. Le premier agent GUI permet de plus à l'utilisateur de spécifier des informations comme le niveau du stock de sécurité ou le temps d'acheminement des marchandises.

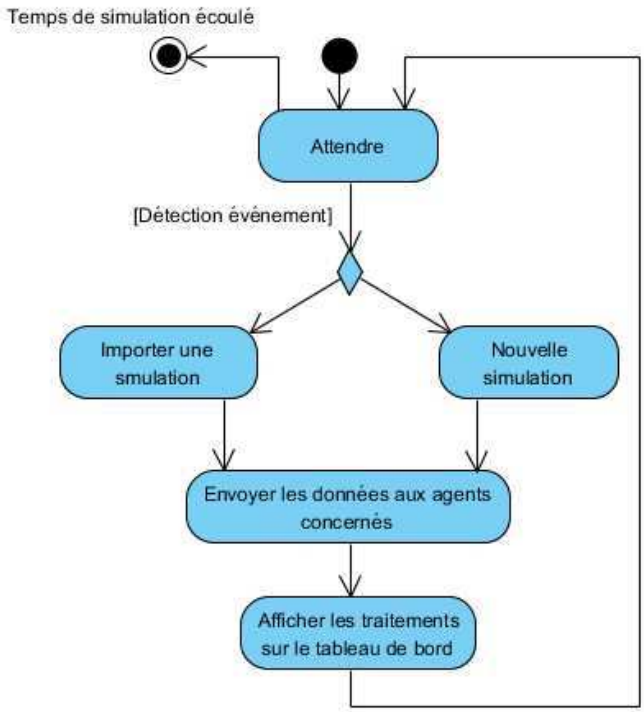

Figure III. 15 Diagramme d'activités de l'agent GUI

L'interaction entre une société d'agents, composée d'agents zones, d'agent NEA et de l'agent Météo est illustrée par le diagramme d'activités à trois partitions de la Figure III.16 suivante :

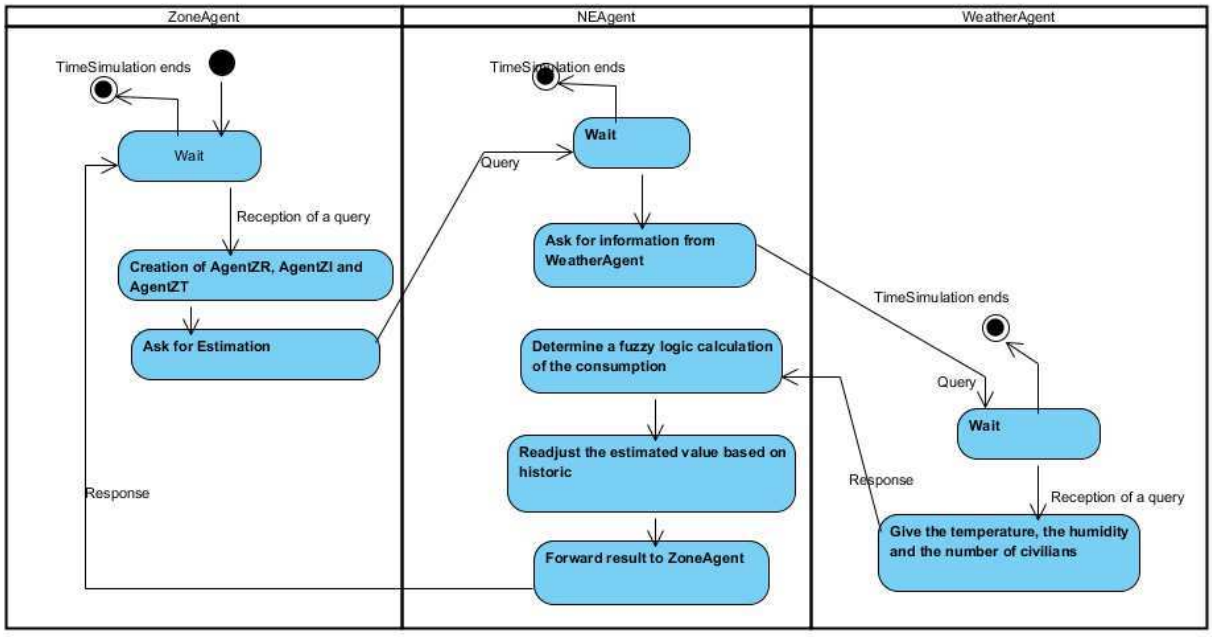

Figure III. 16 Interactions entre les agents

Un exemple de comportements de la société d'agents est illustré par le diagramme de séquence de la figure III.17.

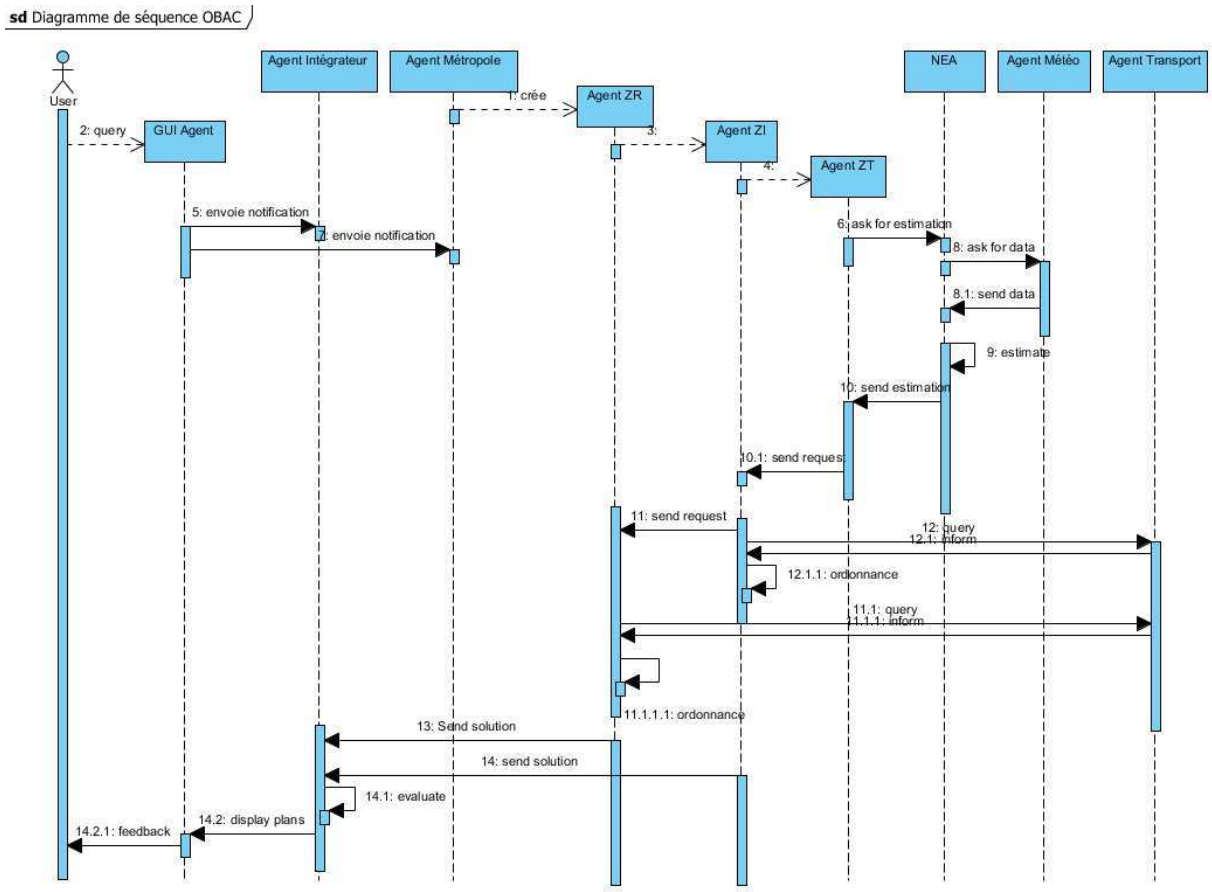

Figure III. 17 Diagramme de séquences

Les comportements détaillés des différents agents, et les différentes méthodes de résolution développées au cœur de chaque acteur, seront décortiqués dans le chapitre IV.

L'ensemble des agents de notre chaîne logistique de gestion de crise, du fait de leurs différentes propriétés (autonomie, communication, coopération,…) peuvent avoir différents modes d'activités. En effet, à un instant *t* donné, certains agents négocient entre eux, d'autres transmettent des commandes alors que d'autres ordonnancent des tâches de livraison… Nous avons dégagé quatre modes d'activités, selon la figure III.18.

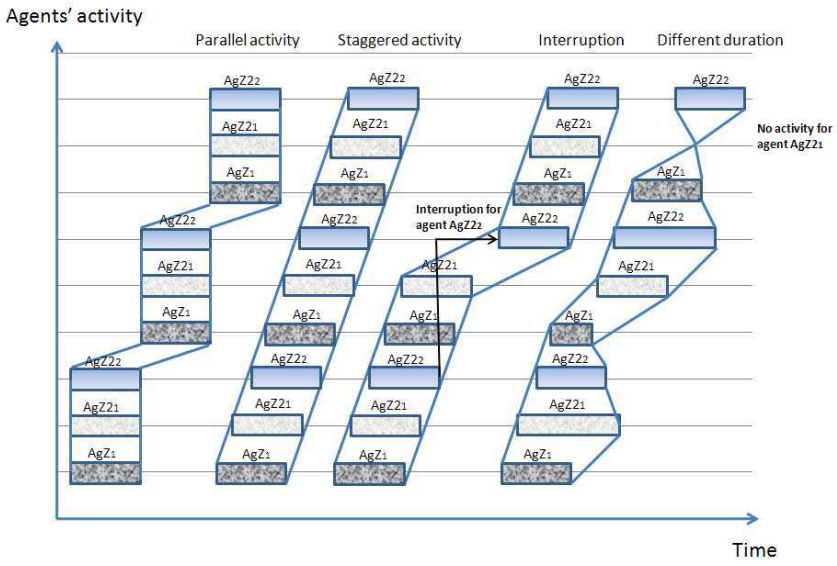

Figure III. 18 Différents modes d'activités des agents

- Activités parallèles: dans cet exemple, trois agents travaillent simultanément, de manière non séquentielle. Un agent attend à la fin d'une activité pour en commencer une autre.
- Activités avec décalage: les trois agents travaillent avec un décalage. Un agent peut démarrer une nouvelle activité même si une première activité est encore en cours.
- Interruption: un agent-zone peut interrompre une activité déjà commencée, s'il reçoit une mise à jour ou une activité plus urgente.
- Différentes durées: les activités des agents sont de durées différentes. Ces durées dépendent du type de l'activité à réaliser (négociation, ordonnancement, estimation…).

III.4.7 Fonctionnement global du système

Chaque jour, quatre grandes étapes permettent de résumer les opérations pouvant s'effectuer:

· Réception des colis.

· Consommation des marchandises.

· Transmission des demandes et besoins au supérieur hiérarchique.

· Envoi des colis à une zone subalterne.

III.3.7.1 Réception des colis

Une première phase consiste à réceptionner les colis reçus par les zones.

L'agent Transport indique ce que chaque zone reçoit. Cette phase est assurée par des messages échangés entre les agents.

L'agent Transport indique, par le biais d'un message, aux agents zones qu'ils ont reçu un colis, en spécifiant le contenu du colis. L'agent zone indique qu'il l'a bien reçu, il met à jour

son stock, et renvoie l'information à l'agent GUI, afin qu'il puisse écrire dans la fenêtre graphique une ligne spécifiant la réception du colis.

III.3.7.2 Consommation des marchandises

Cette deuxième phase a pour but d'enregistrer la consommation des marchandises dans chacune des zones. Cela se traduit par une baisse du stock pour toutes les zones. Chaque agent zone indique la consommation qu'il a enregistrée.

On peut distinguer deux parties dans cette phase.

- La communication au niveau de la chaîne logistique : l'agent zone enregistre la consommation à son niveau, et modifie ses stocks en fonction. Il informe l'agent GUI, toujours afin que les informations soient affichées dans la fenêtre utilisateur.
- La deuxième partie implique l'agent estimateur. En effet, pour fonctionner correctement, l'agent estimateur a besoin d'un retour d'informations sur la consommation ayant réellement eu lieu, afin d'optimiser ses estimations. Ainsi, l'agent zone transmet la consommation réelle à l'agent estimateur. L'information est également transmise à l'agent GUI du NEA, pour que la fenêtre d'estimation puisse rédiger le rapport sur le tableau de bord, et résumant les valeurs optimisées et réelles pour la journée précédente, et pour les différentes ressources considérées.

III.4.7.3 Transmission des demandes

Chaque zone fait une évaluation des besoins qu'elle aura dans les n jours à venir. Ces besoins ne correspondent pas à la consommation journalière ; en effet, le besoin est une demande d'envoi de colis. Ainsi, si la zone s'aperçoit que dans les n jours qui suivent, le stock présent (hors stock de sécurité) ne suffira pas à absorber la consommation ou les colis envoyés aux zones subalternes, alors cette zone commandera à sa zone-mère un réapprovisionnement, d'un montant égal à la quantité nécessaire pour retrouver son seuil de recomplètement.

On part d'un stock actuel, chaque jour, on effectue les modifications suivantes :

· On ajoute les stocks que la zone-mère a déjà envoyé, et qui sont en transit ; ou bien les promesses d'envois formulées par la zone-mère.

· Si la zone est une ZRR ou ZI, on soustrait la quantité de marchandises que l'on prévoit d'envoyer à chacune des zones subalternes ce jour-là.

· Sinon, c'est une zone terminale, auquel cas on soustrait les marchandises correspondant au besoin identifié par les agents estimateurs.

· Si à ce stade, le stock des marchandises que l'on prévoit est inférieur au stock de sécurité, alors cela se traduit par des demandes auprès du supérieur, pour ré-obtenir le seuil de recomplètement. C'est donc à ce stade que l'on définit les besoins d'une zone. Afin de réaliser

une estimation juste des stocks pour les jours suivants, on considèrera dans un premier temps que cette demande sera satisfaite ; ce qui sera confirmé ou infirmé par les promesses d'envoi qui suivront.

L'information des demandes est une information ascendante : les zones terminales déterminent un besoin qu'elles transmettent au supérieur ; ce dernier peut ainsi évaluer ses propres besoins et les transmettre à l'échelon supérieur, et ainsi de suite. C'est lors de la phase suivante, envois des colis, que chaque zone décidera d'envoyer ou non les colis demandés.

III.4.7.4 Envois des colis

La dernière phase est l'envoi des colis ; suite aux demandes formulées par les zones subalternes, chaque zone fait une évaluation des marchandises qu'elle va envoyer ce jour, et qu'elle prévoit d'envoyer dans les jours à venir. C'est particulièrement à cette étape qu'intervient la stratégie logistique à adopter : quoi envoyer, à qui, et à quel moment. Cette stratégie logistique a pour but de définir quelle quantité de marchandises envoyer, en fonction des paramètres suivants : les évaluations des besoins de l'ensemble des zones subalternes pour chaque jour, et la quantité de stock disponible et le stock prévisionnel dans les jours à venir.

Le flux d'envois de marchandises suit un parcours inverse à celui de la précédente phase : l'envoi commence par la zone de regroupement des ressources, qui envoie à l'ensemble de ses zones subalternes des marchandises. Les zones subalternes vont pouvoir mettre à jour leur stock prévisionnel, et envoyer leurs marchandises aux zones subalternes. On a ainsi un flux descendant.

Il faut noter qu'à chaque étape, l'information sur les marchandises envoyées parvient à l'agent GUI, afin que la fenêtre soit mise à jour par rapport aux décisions d'envois.

III.5 Conclusion

Les méthodologies orientées agents s'intéressent à des problèmes de conception de systèmes multi-agents relatifs à : l'identification des agents, la spécification des capacités de raisonnement, l'organisation du système multi-agents et la représentation des interactions entre agents. Elles ont pour objectifs de guider le concepteur dans les phases d'analyse et de conception de systèmes multi-agents.

La modélisation à base d'agents, de chaîne logistique de gestion de crise, offre une représentation du modèle du domaine basée sur le paradigme agents. Cette modélisation repose sur plusieurs modèles permettant de dessiner l'organisation du système multi-agents, la nature des agents, leurs comportements, ainsi que les modes d'interaction.

Nous nous intéressons à un meilleur modèle de déploiement et d'utilisation des agents le long de la CLGC afin de pouvoir disposer de données convenables en minimisant les délais. Pour ce faire, nous proposons dans un premier temps une solution optimisée pour le positionnement des zones logistiques de la CLGC. Ensuite nous proposons différents comportements pour les agents zones, qui seront détaillés dans le chapitre IV.

Chapitre IV : Comportements optimisateurs des agents logistiques

IV.1 Introduction

Plusieurs problématiques sont inhérentes à notre chaîne logistique de gestion de crise et à son fonctionnement. En effet, des décisions ayant de forts enjeux et des répercussions économiques non négligeables doivent être prises. L'investissement dans chacun des maillons de la CLGC est généralement très lourd, il est donc nécessaire de les étudier de façon approfondie, de les concevoir et de les organiser au mieux. On obtiendra ainsi de meilleures performances lors du démarrage ou de la re-conception de la chaîne. Par ailleurs, il existe des interactions entre ces différents maillons ainsi qu'entre des maillons de la chaîne et des éléments extérieurs à celle-ci. Les performances obtenues par chacun et la performance globale de la chaîne dépendent donc de sa phase de conception et des différentes collaborations et coopérations qui existent dans celle-ci. Il s'agit alors de définir au mieux la structure de la CLGC et de prendre les meilleures décisions en fonction du comportement des différents partenaires ou intervenants. Lorsque la structure et le fonctionnement de la chaîne sont définis, en phase d'exploitation, à différents niveaux (stratégique si besoin mais surtout tactique ou opérationnel), il s'agit de planifier et ordonnancer l'acheminement des ressources et la gestion des transports afin de réagir au mieux à la crise.

Nous avons donc adapté et développé un algorithme de positionnement des zones logistiques, qui va permettre d'optimiser la structure de notre CLGC. De la même manière, un processus d'ordonnancement distribué, et donc un nouveau comportement ordonnanceur intégré dans le cœur de chaque agent zone, a été développé et mis en œuvre (Kaddoussi, 2011-b).

Le reste de ce chapitre est organisé comme suit : nous présentons et détaillons dans la section suivante le comportement de l'agent métropole pour résoudre la problématique du positionnement optimisé des zones logistiques. Deux modes seront mis en avant : le mode automatique et le mode manuel. Ensuite, dans la section IV.3, nous nous intéressons au comportement ordonnanceur des agents zones, et plus globalement au processus distribué pour l'optimisation de l'acheminement des ressources, dans le respect des contraintes de délais. Finalement, dans la dernière section, nous présentons une étude de convergence de notre système proposé.

IV.2 Comportement de l'agent Métropole : Positionnement optimisé des zones

Notre CLGC est une chaine dynamique, c'est-à-dire à t= 0, elle n'existe pas encore. Un des plus grands défis de la logistique militaire par rapport à la logistique civile est de garantir de bonnes performances et ceci à chaque nouveau déploiement et malgré la variabilité de l'environnement. Lors d'une crise, nous sommes donc toujours face à une prise de décision quant à la configuration de notre chaine. En effet, il faut décider du nombre de zones qui constitueront la chaine et de leur implantation géographique. Bien entendu, les décisions prises lors de la conception de la chaîne logistique ont un impact sur l'ensemble des décisions prises lors de l'exploitation du réseau logistique.

IV.2.1 Problématique

Plusieurs contraintes interviennent dans la prise de décision relative à l'implantation géographique de la CLGC. En effet, pour assurer une coordination logistique efficace il faut tenir compte de différents points:

- L'importance des dégâts dans le pays touché par le sinistre, qui dépend ici de la position géographique par rapport au centre de la crise. En effet la zone ou le pays le plus touché sera le premier aidé (voir aussi si les dégâts posent des problèmes d'accès: routes détruites, distance des bases logistiques aux victimes trop importante, personnel hospitalier disparu …)
- La politique intérieure du gouvernement (taux de scolarisation, couverture vaccinale importante (exemple : Sri Lanka où la présence de moustiques impliquait un fort besoin en vaccins et autres équipements comme les moustiquaires…), zones tampons empêchant l'accès à certaines zones clés.
- Le climat et la faune locale.
- La probabilité que d'autres catastrophes naturelles interviennent et détruisent les installations.
- La zone géographique (la zone urbaine a plus d'accès aux ressources que la zone rurale, les transports doivent être adaptés à la zone (bateaux, camions civils ou militaires, voie aérienne).

En concertation avec nos experts logisticiens d'EADS, nous avons pu alors dégager les caractéristiques suivantes des bases logistiques intermédiaires (Hubs):

- Proximité géographique aux zones de déploiement ;
- Un climat similaire à celui des zones de déploiement ;

- Infrastructure préexistantes disposant de véhicules de soutien et des installations de réparation de l'équipement ;
- Zone politiquement stable à l'appui de la distribution commerciale.

Nous proposons dans le cadre de nos travaux de recherche d'offrir au décideur logisticien un outil qui le guide dans sa prise de décision au moment de la configuration de la CLGC, pour choisir l'emplacement géographique des bases logistiques intermédiaires (Hubs).

L'objectif ici est de suggérer au concepteur les zones géographiques optimales qui garantissent un acheminement et une réaction rapide et efficace face à la crise. Pour résoudre ce problème, nous proposons un module d'aide à la décision au sein de notre démonstrateur OBAC (Optimisation à Base d'Agents Communicants) qui propose au logisticien, deux modes de fonctionnements. Dans un premier temps, un mode automatique qui propose une surface (qu'on appellera par la suite la surface de Steiner) à l'intérieur de laquelle l'implantation d'un hub est optimisée. Un deuxième mode de fonctionnement a été mis en place, où il revient au logisticien de placer le hub manuellement, en se laissant « conseiller » par le démonstrateur pour optimiser son choix parmi plusieurs zones d'appuis stratégiques préexistantes. Ce deuxième mode de fonctionnement sera privilégié dans les pays où les forces armées disposent déjà de plusieurs zones d'appuis stratégiques préinstallées. On privilégie dans ce cas l'utilisation de ces zones comme zones intermédiaires, au lieu d'en créer des nouvelles (le cas du premier mode automatique).

Nous présentons dans la sous-section suivante le problème de Steiner pour la création de zones intermédiaires, ensuite nous détaillons le processus d'aide à la décision pour l'implantation géographique des Hubs.

IV.2.2 Problème de Steiner

Le problème de l'arbre de Steiner (nommé en référence au mathématicien Jakob Steiner) est un problème très classique de la théorie des graphes. C'est un problème d'optimisation combinatoire relativement proche du problème de l'arbre couvrant minimal.

Problème de l'arbre couvrant minimum	Problème de Steiner
- Données : graphe non orienté, longueur d_{ij} associée à chaque arc (i,j) ; - Problème : déterminer un arbre de poids minimum pour connecter tous les sommets du graphe.	- Données : graphe non orienté, longueur d_{ij} associée à chaque arc (i,j), ensemble de sommets terminaux ; - Problème : déterminer un arbre de poids minimum pour connecter tous les sommets terminaux.

Tableau IV. 1 Problème de l'arbre couvrant minimum et problème de Steiner

Dans les deux problèmes, étant donné un ensemble V de sommets, il s'agit de trouver un arbre A reliant tous les sommets de V. Alors que dans le problème de l'arbre couvrant minimal, tous les sommets de l'arbre A doivent être dans V, il est autorisé dans le problème de l'arbre de Steiner d'utiliser des points en dehors de V. Dans les deux problèmes, chaque arête a un coût donné. Le coût de l'arbre étant donné par la somme du coût de ses arêtes, il s'agit de trouver l'arbre de coût minimal.

IV.2.2.1 Problème de Steiner dans les graphes

Soit un graphe non orienté G = (V, E), avec V : sommets, E : arêtes, L : terminaux (obligatoires) et c_e : poids de l'arête c ∈ E.

L'objectif est d'obtenir un arbre couvrant des sommets terminaux L de poids minimum (cas particulier : si L=V, problème de l'arbre couvrant de poids minimum).

Il existe différentes contraintes que l'on peut appliquer à l'arbre. La plupart des problèmes sont NP-complets (donc considérés comme difficiles à calculer). Quelques cas de figures sont résolubles en un temps polynomial. Pour les autres cas, la résolution se fait *via* une recherche heuristique.

Ce problème de Steiner a trouvé de multiples applications dans un panel de disciplines très étendu, comme les projets des réseaux d'ordinateurs et de télécommunications, les problèmes de la phylogénie en biologie, conception de grands réseaux (télécommunication, distribution électrique, pipelines,. . .)

Exemples :

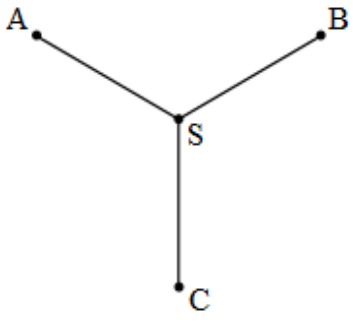

Figure IV. 1 Solution pour trois points.
Le point de Steiner S
est au centre des trois autres points A, B et C.

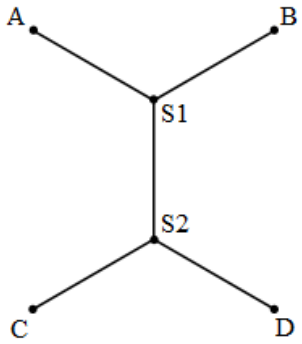

Figure IV. 2 Solution pour quatre points.
Dans cet arbre, il y a deux points de Steiner : S1 et S2

La diminution de la complexité liée à la construction de l'arbre minimal couvrant un groupe de nœuds dans un graphe connexe non-orienté a fait l'objet de nombreux travaux.

IV.2.2.2 Etat de l'art

La problématique de construction de l'arbre minimal de Steiner a longtemps été traitée dans la littérature. Le temps de calcul demandé par les solutions connues dépend d'une façon exponentielle de la taille du problème (nombre de nœuds dans le groupe multipoints et dimensions du graphe). Jusqu'à présent, la construction exacte de l'arbre pour une cinquantaine de nœuds a pu être réalisée. Dans les cas réels, on utilise des heuristiques fournissant des solutions approchées de l'arbre minimal de Steiner. Une des heuristiques les plus simples est l'algorithme de Takahashi et Matsuyama (Takahashi et al., 1980) qui reprend l'idée de Prim pour construire un arbre couvrant. L'algorithme consiste à choisir arbitrairement un sommet et à faire croître un arbre à partir de ce sommet. Chaque augmentation se fait de la manière la plus économique possible.

Principe : L'étape d'initialisation consiste à choisir un sommet quelconque du graphe initial. Au bout de la première étape, on se retrouve ainsi avec un arbre contenant 1 sommet et 0 arête. Ensuite, on construit récursivement l'arbre minimal de la façon suivante : à l'étape n, ayant déjà construit un arbre contenant n sommets et n-1 arêtes, on établit la liste de toutes les arêtes reliant un sommet de l'arbre à un sommet qui n'est pas sur l'arbre. On choisit alors une arête de poids minimal, que l'on rajoute à l'arbre ; l'arbre contient à présent n+1 sommets et n arêtes. L'algorithme se termine lorsque tous les sommets du graphe sont contenus dans l'arbre.
La complexité de l'algorithme dépend fortement de la manière dont est implémenté le choix de l'arête/sommet à ajouter dans l'ensemble à chaque étape. Avec une représentation naïve, on obtient une complexité en O($a * S$) avec a le nombre d'arêtes et S le nombre de sommets.

Une autre heuristique fournissant des solutions approchées de l'arbre minimal de Steiner est l'algorithme de Kruskal (Kruskal, 1956). C'est un algorithme qui permet de paralléliser l'algorithme de Djikistra. C'est donc un bon moyen d'obtenir un arbre couvrant minimum (arbre de taille minimum ne reliant que des points de P). Cet algorithme considère simultanément tous les membres du groupe. Il propose d'initialiser une forêt (un ensemble d'arbres) avec les membres du groupe (chacun des nœuds initialise un arbre isolé). Son algorithme permet d'obtenir un seul arbre couvrant à l'aide des connexions successives entre les arbres isolés. Les connexions sont réalisées à l'aide des plus courts chemins.

Principe : Le squelette de l'algorithme est simple : l'idée est d'initialiser l'ensemble P comme une forêt, c'est-à-dire comme un ensemble de N arbres de 1 point chacun. A l'aide d'une matrice des distances entre les arbres de P, on détermine ensuite les deux arbres les plus près l'un de l'autre et on les unifie. On recommence le fonctionnement jusqu'à unification totale

de la forêt. La complexité de l'algorithme, dominée par l'étape de tri des arêtes, est $\Theta(a \log V)$ avec *a* le nombre d'arêtes et *V* le nombre de sommets du graphe G.
Zelikovsky (Zelikovsky, 1993) propose un algorithme qui permet de construire un arbre dont le quotient *coût du résultat / coût de l'arbre minimum de Steiner* est inférieur à 11/6. Zelikovsky utilise (à la place des plus courts chemins) des arbres pour couvrir des triplets de nœuds.

Selon la métrique utilisée pour le calcul de la distance entre les sommets dans V, deux versions de l'arbre de Steiner ont été étudiées :

- Graphe de Steiner minimal (Graph Steiner Minimal Tree : SMT): Dans cette version, l'ensemble des sommets et métrique est donné par un graphe fini.

- Arbre de Steiner euclidien minimal (euclidean SMT): Dans cette version, l'ensemble V représente tout l'espace euclidien et est donc infini. Généralement la métrique est donnée par la distance euclidienne. Pour deux points de coordonnées (x_1, y_1) et (x_2, y_2), la distance est :

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

  Dans certaines applications, telles que le routage VLSI (Very Large Scale Integration routing), la norme L1, également connu sous le nom distance rectiligne, est utilisée. La distance est définie alors par :

$$|x_1 - x_2| + |y_1 - y_2|$$

  La figure suivante illustre un arbre de Steiner euclidien minimal et un arbre de Steiner minimal pour les graphes.

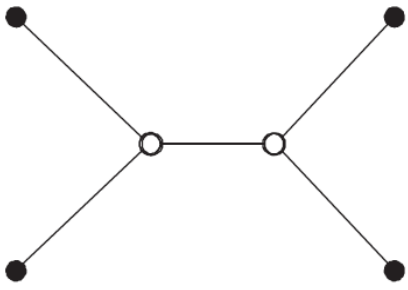

Figure IV. 3 Arbre de Steiner euclidien minimal

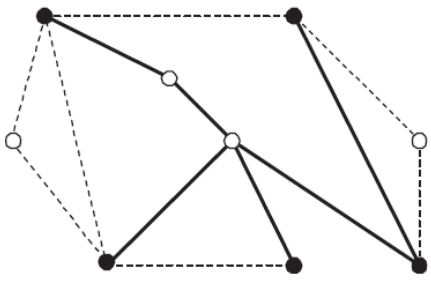

Figure IV. 4 Graphe de Steiner minimal

L'approche générique pour trouver l'arbre de Steiner est la suivante :

Approche générique
In: P
Out: A
1: **begin**
2: A= Ø
3: **While** (A n'est pas un AC)
4: Trouver arête (x,y) de poids minimum qui peut être ajouté à A
5 A:= A+ {(x,y)}
6:**Return** A
7: **end**

Avec AC : Arbre Couvrant

En effet, une méthode bien connue pour se rapprocher d'un SMT (*Steiner Minimal Tree*) est d'utiliser un arbre couvrant minimal (*Minimal Spanning Tree : MST*). On construit d'abord un graphe complet avec L sommets et poids des arêtes égales à la longueur du plus court chemin. Puis on trouve un MST, dans lequel chaque arête correspond à un plus court chemin sur le graphe original. Enfin, le MST est transformé à nouveau en un arbre de Steiner en remplaçant chaque arête par le chemin le plus court et en effectuant un post-traitement simple pour supprimer les cycles possibles.

Algorithm : MST-STEINER
Input : A graph $G = (V,E,w)$ and a terminal set $L \subset V$.
Output: A Steiner tree T.
1: Construct the metric closure G_L on the terminal set L.
2: Find an MST T_L on G_L.
3: T $\leftarrow \varnothing$.
4: **For** each edge $e= (u,v) \in E(T_L)$ **do**
4.1: Find a shortest path P from u to v on G.
4.2: **If** P contains less than two vertices in T **then**
Add P to T;
Else
Let p_i and p_j be the first and the last vertices already in T;
Add subpaths from u to p_i and from p_j to v to T.
5: Output T.

Figure IV. 5 Algorithme général de Steiner

IV.2.3 Algorithme de positionnement des zones

Notre module pour le positionnement des zones propose à l'utilisateur deux modes de fonctionnement. Dans le premier mode, qualifié de mode automatique, nous déterminons un arbre de Steiner approché. Les zones intermédiaires ou Hubs susceptibles d'être implantées géographiquement dans la CLGC appartiennent à des surfaces de la forme de cercles, de

centres les points de Steiner, et de rayons réglables par le logisticien. A l'intérieur de ces surfaces, tout placement de zone est considéré optimisé. Nous appellerons cette surface : surface de Steiner. Le rayon du cercle est une valeur qui correspond à une relaxation que nous autorisons et qui permet de contourner des contraintes qui rendent impossible l'implantation de la zone exactement sur le point de Steiner trouvé (sans pour autant trop dégrader la solution).

Dans le deuxième mode, nous proposons à l'utilisateur un outil d'aide à la décision pour choisir, parmi un ensemble de zones d'appuis stratégiques préexistantes, la zone géographique et plus particulièrement la ville au niveau de laquelle il serait préférable d'installer la zone logistique. Cette étape va permettre de prendre en considération différents paramètres supplémentaires fixés par les experts logisticiens, dans le choix de la localisation de la zone. Nous montrons que cette approche permet la prise en compte de l'expertise humaine et de facteurs géographiques, politiques et économiques.

IV.2.3.1 Algorithme de Steiner : vers la création dynamique de la CLGC

Nous travaillons sur un ensemble donné P constitué de N points.

$P = \{P_1, P_2, \ldots, P_N\}$

Chaque point est entièrement caractérisé par ses coordonnées. Ces coordonnées représentent respectivement la latitude et la longitude du point.

$$P_i = (u_i, v_i)$$

Le problème consiste à relier les points de P en utilisant si nécessaires des points extérieurs à P. Ces points sont ce qu'on appelle des points de Steiner et ils appartiennent à l'ensemble S. La théorie de Steiner montre que l'ensemble S est de cardinalité maximale N-2.

$$S = \{S_1, S_2, \ldots, S_{N-2}\}$$

La difficulté du problème réside dans le nombre et la position des points de S, et donc dans la recherche des coordonnées de chacun des points de S. La construction de ces points de S nécessite l'utilisation de points intermédiaires X_i.

$$S_i = (u_i, v_i), \quad X_i = (x_i, y_i)$$

L'algorithme de Steiner retourne une zone surfacique sous forme d'un cercle, et de rayon R réglable, au cours de la crise, pour agrandir ou diminuer la surface optimisée selon la situation sur le terrain. Ce paramètre est contrôlé par le logisticien en se basant sur les historiques des opérations de déploiement en cas de crise.

Heuristique proposée :

On commence de la même façon que pour Kruskal : on initialise une forêt et une matrice des distances. On sélectionne ensuite la représentation des deux arbres les plus proches et on les réunifie avec la méthode de Zelikovsky. On recommence ensuite sur la nouvelle forêt jusqu'à unification totale.

Vu qu'on recherche à contraindre la complexité à $O(N^2)$ on va se limiter dans le choix des représentations. Ces dernières seront :

- Les points S construits lors de la dernière réunification,
- Les extrémités du chemin utilisé si la réunification n'a pas construit de points S,
- Le point P s'il est isolé,

De cette façon à chaque instant chaque arbre de la forêt est représenté par 2 points au maximum. Ainsi on ne va construire que des arbres de Steiner à 4, 3 ou 2 points.

Structure générale

```
/*Comporte une initialisation et une simple boucle de parcours.*/

Initialisation des N points et de la matrice des distances ;

Jusqu'à la fin faire
        Considérer les arbres les plus proches ;
        Réunifier les arbres ;
        Actualiser la matrice des distances ;
Fin Tant que
```

Figure IV. 6 Algorithme général

- *Actualiser les distances pour la matrice*

Le calcul ne se fait pas comme dans l'algorithme de Kruskal par la méthode du plus court chemin. Il se fait comme dans l'algorithme de Zelikovski. C'est-à-dire qu'on considère la distance entre deux arbres comme la somme des longueurs de l'arbre de Steiner qui permettrait sa réunification, à laquelle on retire la longueur des boucles. Finalement pour calculer la distance qui sépare l'arbre nouvellement créé des anciens arbres de la forêt, nous construisons l'arbre de Steiner qui nous servira lors de la prochaine phase de réunification. Ce qui explique pourquoi cette phase de réunification n'est qu'une initialisation des représentations pour les prochaines itérations.

Actualiser les distances pour la matrice

```
/*Comporte une initialisation et une simple boucle de parcours.*/

Pour chaque arbre de la forêt faire :
        Si on a un seul représentant par arbre
                Simple segment ;
                Pas de boucle à supprimer ;

        Si un des arbres à deux représentants
                Arbre de Steiner 1-S ;
                On élimine 1 ou 0 boucle ;

        Sinon
                Arbre de Steiner 2-S ;
                On élimine 1,2 ou 0 boucles ;

        Si valeur distance calculée plus petite alors garder arbre ;
Fin Faire
```

Figure IV. 7 Algorithme pour Actualiser les distances

Réunifier les arbres

```
/*Se fait simplement car l'étape précédente dans le code considère
les arbres les plus proches déterminés dans le calcul des
distances*/

Si on a un seul représentant par arbre
        Sommets du segment deviennent la représentation ;
Si un des arbres à deux représentants
        Le point S devient la représentation ;
Sinon
        Les 2 points S deviennent la représentation ;
```

Figure IV. 8 Algorithme pour Réunifier les arbres

- *Construction de l'arbre 1-S*

Géométriquement, cet arbre de Steiner est le plus simple. Il suffit alors de placer correctement le point S. Pour cela on distingue différents cas, et on appelle les sommets du triangle formé par des représentations A, B et C. On utilise ensuite la réduction de Melzak (Melzak, 1961) pour placer le point S.

Construction de l'arbre 1-S

```
/*Se fait simplement car l'étape précédente dans le code considère
les arbres les plus proches déterminés dans le calcul des
distances*/
```

Sélection de la plus grande distance du triangle ;
Construction du point X ;
Construction du cercle circonscrit;
Condition d'alignement et d'intersection avec le cercle ;
Détermination de $\{u_i,v_i\} = S_i$;

Figure IV. 9 Algorithme de construction de l'arbre 1-S

- *Construction de l'arbre 2-S*

On utilise encore une fois la réduction de Melzak. Nous partons d'un ensemble de 4 points A, B, C et D. Nous construisons 2 couples de points X, soit 4 points. Nous mesurons ensuite les distances entre ces deux couples et on ne gardera que le cas le plus favorable.

Construction de l'arbre 2-S

Construction de x1 et de x2 pour la longueur AB ;
Construction de x3 et de x4 pour la longueur CD ;
Mesure de 1-3, 1-4, 2-3 puis 2-4 ;
On garde l'association la plus favorable ;
Construction des cercles circonscrits;
Condition d'alignement et d'intersection avec le cercle ;
Détermination de $\{u_i,v_i\} = S_i$;

Figure IV. 10 Algorithme de construction de l'arbre 2-S

IV.2.3.2 Aide au positionnement : le mode discret

L'objectif de ce mode de fonctionnement est d'améliorer le soutien logistique des forces déployées à l'étranger, en facilitant la gestion des transports et la distribution des équipements aux endroits stratégiques du globe, et en favorisant le placement des bases logistiques intermédiaires à des lieux stratégiques.

Une brève étude sur le problème de l'emplacement des installations et leurs domaines d'application (le transport commercial, les télécommunications et les applications militaires) est donnée dans (Ghanmi et al., 2008). Une revue de la littérature sur les modèles discrets pour l'emplacement des installations, décrivant certaines contraintes telles que la capacité des installations, l'emplacement statique/dynamique des installations, peuvent être trouvés dans (Sibel et al., 2008).

L'approche repose sur la mise en place d'une série de plates-formes permanentes et temporaires de soutien opérationnel à des endroits clés partout dans le monde.

Le choix de l'emplacement d'un Hub doit répondre aux conditions suivantes :

- Proximité géographique aux zones de déploiement ;
- Un climat similaire à celui des zones de déploiement ;
- Infrastructure préexistantes disposant de véhicules de soutien et des installations de réparation de l'équipement ;
- Zone politiquement stable à l'appui de la distribution commerciale.

En se basant sur les critères précédents et en utilisant un ensemble d'indicateurs politico-éco-militaires, les forces armées ont pu identifier des régions d'appui stratégique dans le monde, qui seront favorisées lors de l'implantation des zones logistiques.

Disposant de *n* zones d'appuis stratégiques et d'une liste *m* de destinations de déploiement possibles, le modèle d'optimisation du placement des bases logistiques proposé doit donner:

- L'emplacement (discret) des installations parmi les régions d'appuis stratégiques existantes ;
- Le nombre d'installations.

Les régions d'appuis stratégiques sont évaluées selon les valeurs des variables de décision suivantes :

- h_i = nombre de personnel disponible / BL_i (Base Logistique);
- d_{ij} = distance de la BL *i* (et de la Métropole (i=0)) à la zone *j* ;
- N_{ij} = réseau routier, nombre de routes disponibles pour aller de la BL *i* vers la zone *j* ;
- P_{ij} = nombre de ports disponibles ;
- A_{ij} = nombre d'aéroports ;
- C_{ij} = réseau de communication ;

Ces variables sont soumises à certaines contraintes :

- Contrainte de distance : $d_{ij} \leq K$ (K : distance en km fixée par EADS) ;
- Contrainte de transport : $N_{ij} \leq 0$ (présence de liaison terrestre) ;

L'algorithme proposé calcule un indice de pertinence de chacune des zones d'appuis, à partir des valeurs des variables ci-dessus. L'indice de pertinence est le résultat d'une pondération entre toutes ces variables. Le logisticien aura la possibilité de donner plus d'importance à un critère donné, en augmentant sa pondération.

IV.3 Comportement de l'agent zone : ordonnancement des tâches de livraison

Nous nous proposons d'introduire au sein des agents zones les techniques d'optimisation et d'ordonnancement plus particulièrement afin de leur permettre de faciliter la circulation des

flux de la CLGC et déterminer une meilleure solution d'ordonnancement satisfaisant un certain nombre de critères.

Les approches centralisées classiques proposent de concentrer la résolution du problème d'optimisation au sein d'un unique agent qui joue le rôle de médiateur. Tous les agents doivent alors communiquer leurs préférences à cet agent qui va se charger de trouver la solution optimale. L'avantage de cette approche réside dans l'optimalité des solutions trouvées et dans la simplicité de la communication entre agents. Seulement, cette centralisation de la résolution pose problèmes aux niveaux de la tolérance aux pannes et à l'adaptabilité. En effet, une perturbation ou un changement mineur dans les données du problème nécessite le redémarrage du processus de résolution.

Nous proposons, dans ce contexte, une approche de résolution distribuée et coopérative faisant interagir l'ensemble des entités composant notre système. Le problème d'ordonnancement considéré va permettre d'établir des plans d'acheminement des ressources le long de la CLGC.

IV.3.1 Définitions

Résoudre un problème d'ordonnancement revient à organiser la réalisation d'un ensemble d'opérations élémentaires (les tâches), en exploitant les capacités des machines disponibles (les ressources). Cette réalisation doit respecter un ensemble de règles techniques (les contraintes), tout en présentant le maximum d'efficacité (selon les critères d'évaluation choisies) (Artiba, 1990 ; Carlier et Chrétienne, 1988). L'ordonnancement vise à améliorer l'efficacité d'une entreprise en minimisant les coûts et en respectant les délais de livraison. Les problèmes d'ordonnancement apparaissent dans tous les domaines : informatique, militaire, industrie, transport, construction, administration, etc.

Les éléments principaux constituant un problème d'ordonnancement sont alors les suivants : les tâches (ou opérations), les ressources, les contraintes et les critères d'optimisation.

IV.3.2 Caractérisation d'un problème d'ordonnancement

Les problèmes d'ordonnancement consistent à organiser plusieurs tâches en respectant certaines contraintes et en optimisant certains critères.

IV.3.2.1 Les tâches

Ce sont les opérations à effectuer. Les auteurs classent les tâches en deux catégories : les tâches préemptives où une action peut être interrompue pour être continuée ultérieurement, et

les tâches non préemptives où l'exécution de l'opération ne peut pas être interrompue avant qu'elle ne soit achevée.

Chaque tâche ou chaque job peut être caractérisé par une date de début au plus tôt (*release date*) ; une durée pour la tâche (*processing time*) ; une date de fin souhaitée (*due date*) ; une date de fin obligatoire (*deadline*) et un poids relatif (*weight*) ou importance.

IV.3.2.2 Les ressources

L'exécution de différentes opérations et tâches nécessite la mise en œuvre d'un ensemble de moyens et d'opérateurs. Cet ensemble représente donc les ressources indispensables au bon fonctionnement du système (fabrication, distribution,…). Une ressource est généralement disponible en quantité limitée.

Nous en distinguons deux types : les ressources renouvelables et les ressources non renouvelables. Une ressource est renouvelable si après avoir été allouée à une ou plusieurs tâches, elle reste disponible pour effectuer l'exécution de certaines tâches (le personnel, les machines, les moyens de transports,…). Dans le cas contraire, elle est non renouvelable ou consommable (matières premières, budget) ; et sa disponibilité est limitée par son utilisation pour réaliser certaines tâches.

Qu'elle soit renouvelable ou consommable, la disponibilité d'une ressource peut varier au cours du temps. Sa courbe de disponibilité est en général connue a priori, sauf dans les cas où elle dépend du placement de certaines tâches génératrices.

IV.3.2.3 Les critères

Parmi un ensemble d'ordonnancements possibles, nous sommes amenés à choisir la solution la plus satisfaisante. Cette notion de satisfaction dépend directement du critère préalablement défini. Ces critères servent donc à l'évaluation et à la comparaison des différents ordonnancements possibles.

Les critères d'optimisation s'expriment généralement en fonction des dates de fin des tâches (ou jobs) C_j (completion times).

Les critères sont généralement exprimés en fonction des mesures suivantes :

- le retard algébrique $L_j = C_j - d_j$ (lateness) ;
- le retard absolu $T_j = max(0, C_j - d_j)$ (tardiness) ;
- l'avance $E_j = max(0, C_j - d_j)$ (earliness);
- le pénalité unitaire de retard $U_j = 0$ *si* $C_j \leq d_j$, $U_j = 1$ sinon ;
- la durée de séjour dans l'atelier $F_j = C_j - r_j$;
- une fonction générique $f_j(t)$ donnant le coût induit si on termine J_j à t.

On cherche alors à minimiser un ou plusieurs critères F(C1, . . . ,Cn)

- une fonction générique de coût maximum
- $f_{max} = max_j\{f_j(C_j)\}$
- la durée totale de l'ordonnancement $C_{max} = max_j\{C_j\}$
- le retard algébrique maximum $L_{max} = max_j\ L_j$
- le retard maximum $T_{max} = max_j\ T_j$

IV.3.2.4 Les contraintes

Il s'agit de conditions à respecter dans la construction de l'ordonnancement pour qu'il soit réalisable. Les contraintes expriment une restriction sur les domaines de valeur de variables ; Une classification des contraintes est possible selon la disponibilité des ressources et suivant l'évolution temporelle (Carlier et al, 1988) :

- Les contraintes de ressources

  Plusieurs types de contraintes selon la nature des ressources. Les contraintes relatives aux ressources peuvent être disjonctives ou cumulatives :

  - *Contraintes cumulatives* : L'intervention des divers moyens (main d'œuvre, matériels …) impose à l'ordonnancement des contraintes de type potentiel. Les matériels ou les machines alloués au projet à ordonnancer ont des capacités fixées par leurs performances. La main d'œuvre disponible est en général limitée, impliquant la limitation du nombre de tâches à réaliser en parallèle.

  - *Contraintes disjonctives* : L'hypothèse disjonctive induit une contrainte de réalisation de tâches sur des intervalles temporels disjoints pour une même ressource. Elles expriment le fait que deux tâches ne peuvent avoir lieu en même temps sans que l'on puisse dire laquelle doit être effectuée avant l'autre.

  Considérons deux tâches *i* et *j,* et supposons que leur exécution nécessite l'emploi d'un matériel unique (une grue, une machine…).Les deux tâches ne pourront donc être réalisées en même temps; en d'autres termes, les intervalles de temps : $(t_i,\ t_i+k_i)$ et $(t_j,\ t_j+k_j)$ ne peuvent pas avoir une partie commune. Les deux tâches *i* et *j* seront dites en disjonction .Une telle contrainte peut être formulée à l'aide de deux inégalités de potentiel: $t_j - t_i \geq k_i$

- Les contraintes temporelles

  Ces contraintes représentent des restrictions sur les valeurs que peuvent prendre certaines variables temporelles :

- *Contraintes de succession temporelle* : Elles expriment la relation d'antériorité entre les tâches, une telle tâche ne peut commencer avant la fin d'une autre.

Considérons deux taches *i* et *j* dont les dates de début sont respectivement t_i et t_j.

Dire qu'il existe une contrainte de succession entre les tâches *i* et *j* consiste à limiter soit inférieurement soit supérieurement l'intervalle de temps qui peut s'écouler entre t_i et t_j, l'intervalle de temps qui sépare t_i et t_j va être donc borné inférieurement par une quantité que nous noterons a_{ij} : $t_j - t_i \geq a_{ij}$

- *Contraintes des dates dues* : Elles expriment la localisation des tâches dans le temps.

Une tâche ***i*** ne peut être débutée avant une date fixée $\boldsymbol{c_i}$ **;** par exemple ; une livraison ou une décision administrative.

$$t_i \geq c_i$$

ou une tache ***i*** doit être terminée avant une date fixée $\boldsymbol{c_i}$; par exemple ; un engagement commercial.

$$t_i \leq c_i$$

ou enfin, une tache ***i*** doit être impérativement terminée à une date fixée $\boldsymbol{c_i}$; sans avance ni retard. Cette contrainte est la combinaison des deux contraintes précédentes.

$$ti \geq ci$$

$$ti \leq ci$$

ce qui implique: $ti = ci$

IV.3.3 Résolution centralisée/distribuée des problèmes

Dans un contexte réel d'application, il est souvent préférable d'opter pour le choix d'une solution optimisée demeurant faisable face à des perturbations, au lieu d'une solution optimale qui devient rapidement obsolète lors de l'exécution de l'ordonnancement.

Globalement, les systèmes d'ordonnancement les plus répandus se basent sur une politique centralisée de résolution (qu'elle soit en-ligne ou hors-ligne) des problèmes d'ordonnancement. En effet, le processus de résolution des problèmes d'ordonnancement est généralement ramené à un problème de décision global car il s'agit souvent de gérer l'organisation de la totalité des ressources. Cette organisation centralisée des décisions sous-entend alors une connaissance globale des paramètres du système et l'existence d'une entité globale qui supervise les activités des ressources, et qui prend des décisions globales (devant être respectées par l'ensemble des ressources qu'elle gère).

Le point fort de l'approche centralisée réside dans la cohérence globale des décisions prises. Son inconvénient est qu'elle exige une transparence totale des différents acteurs, alors que

supposer qu'une entité dispose d'une vision globale de tout le système est parfois non réaliste. Le deuxième inconvénient est qu'elle ne permet pas d'ajustement des décisions prises globalement.

Dans de nombreux champs d'application (chaînes logistiques, projets industriels, etc.), les ressources utilisées sont souvent réparties au sein d'un ensemble d'acteurs se trouvant parfois en situation de concurrence et disposant de fait, de leur propre autonomie de décision et d'objectifs propres dont il est fondamentale de tenir compte. Ce type d'organisation constitue précisément le contexte où les méthodes de résolution distribuée peuvent être profitables dans la mesure où chaque acteur doit faire face non seulement aux spécificités de sa propre organisation et de son propre environnement local.

Il est alors intéressant de considérer l'ordonnancement sous l'hypothèse d'une fonction distribuée (décisions d'ordonnancement distribuées entre les différents acteurs) où la solution globale résulte d'une coopération entre les différents acteurs, chacun gérant son propre ordonnancement local robuste. Le résultat de la coopération devant converger vers un compromis satisfaisant les performances locales ainsi que la performance globale.

Dans le cadre des chaînes logistiques, plusieurs travaux de recherches font l'hypothèse que les décisions sont distribuées au sein d'un réseau de centres de décision interdépendants, et proposent des mécanismes d'aide à la décision pour la coopération. Le secteur industriel compte aussi de plus en plus de logiciels de gestion de chaînes logistiques offrant des solutions basées sur différents aspects d'organisation coopérative. Toutefois, peu de travaux de recherche se sont intéressés à l'ordonnancement distribué et coopératif. La majorité des travaux proposent des approches basées sur les Systèmes Multi-Agents (SMA).

IV.3.4 Méthodes de résolutions distribuées

Différentes méthodes ont été développées pour résoudre la problématique des plans d'ateliers distribués. Dans (Lee et al., 2002), le modèle de la chaîne logistique traitée est représenté par des modèles mathématiques. Cependant, l'environnement incertain, les perturbations et les différents aléas qui entourent la chaîne logistique rendent ces modélisations incomplètes et les solutions fournies pas souvent acceptables. Dans (Luder et al., 04 ; Gupta et al., 02 ; Kubota et al., 99), des simulations ont permis l'évaluation de la CL en la considérant comme une seule entreprise centralisée. Néanmoins des difficultés liées au nombre important d'entités à modéliser et au niveau de détail désiré lors d'une modélisation en équipe de travail forment les premières limites de cette approche. D'autres concernent la quantité d'événements à simuler, la puissance de calcul des ordinateurs, la réutilisation des modèles de simulation et la

protection de la propriété intellectuelle. Finalement, la modélisation de tous les sites d'une CL demeure possible mais la simulation sur un seul processeur n'est pas toujours réalisable. De plus, il est difficile d'obtenir une évaluation fine de la faisabilité en se basant sur un modèle agrégé. L'évaluation de la faisabilité de la CL ne se ramène pas non plus à la somme des évaluations de la faisabilité des plans de chaque site. Compte tenu de ces limites, la validation d'ordonnancement multi-site nécessite la définition d'une architecture de simulation distribuée.

Différents travaux ont été menés sur la simulation distribuée de grands systèmes de production. Un premier atout en faveur de la décentralisation concerne la protection des informations de chaque acteur de la CL. Ceux-ci peuvent choisir de masquer certaines données lors de la simulation distribuée. Les modèles distribués sont construits et maintenus localement et ne se rejoignent que lors de l'évaluation (Mertins et al., 2005). De plus, la performance globale du temps de simulation est améliorée grâce à la distribution (Turner et al., 2000). Pour synchroniser des simulateurs au sein d'une grande simulation, le protocole HLA (High Level Architecture) a été proposé par le DMSO (Defense Modeling and Simulation Office). Il facilite l'interopérabilité et la réutilisation des simulations. Il a pour intérêt de réduire les coûts de la modélisation et de la simulation et de répondre aux besoins de grandes simulations faisant intervenir des composants géographiquement distribués. HLA met en œuvre des algorithmes pour la synchronisation et le respect de la chronologie pour les événements simulés et notamment celui défini dans (Chandy et Misra, 1978). HLA permet de synchroniser une fédération, i.e, un ensemble de fédérés partageant un modèle objet commun, le FOM (Federation Object Model), contenant toutes les informations relatives à l'exécution de la simulation. Un fédéré est un composant de la fédération comprenant un simulateur auquel peuvent être associés un opérateur, une machine ou un atelier complet. Le RTI (Run-Time Infrastructure) constitue une implantation informatique des spécifications d'interface HLA. Il s'agit d'un processus informatique assurant les communications entre les fédérés d'une même fédération en offrant les services de HLA pour la synchronisation et la chronologie des événements.

IV.3.5 Positionnement de notre étude

Après l'étude bibliographique sur les travaux réalisés dans le domaine de la CL, nous avons constaté que la majorité des recherches sont basées sur des modèles centralisés ne tenant pas compte de ce qui se passe dans chaque unité organisationnelle de la chaîne.

L’enjeu majeur pour notre chaîne logistique est la livraison. Il faut veiller à limiter les retards de livraison dans les zones sensibles en synchronisant correctement les différentes zones. En cas d’erreurs ou de délais trop importants, les dommages engendrés peuvent être très coûteux voir même catastrophiques.

Notre problème d’ordonnancement est composé, dans sa représentation la plus complète, de plusieurs entités, comme illustré dans la figure suivante :

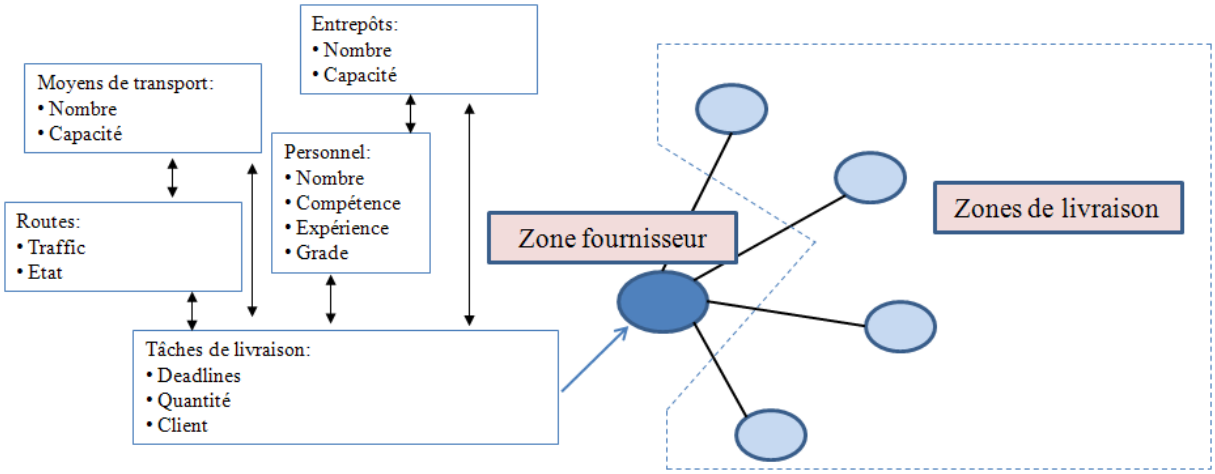

Figure IV. 11 Modèle type d’un système d’ordonnancement de livraison

- *Les tâches de livraison* : Représentent une quantité de ressources à délivrer au sein d’une autre zone, sous certaines contraintes. Cette distribution a un coût variable selon la distance entre le fournisseur et l’utilisateur. Les temps de chargement et de déchargement sont inclus dans le délai de livraison.
- *Les entrepôts* : Nous disposons d’entrepôts dans chaque zone afin de stocker la marchandise (nourriture, eau, médicaments, vêtements etc.).
- *Les moyens de transport* : sont les moyens de transport à utiliser pour l’acheminement des ressources. Le transport des biens peut se faire par des camions, des bateaux ou des avions (avec une prépondérance de l’utilisation de la voie terrestre). Le coût et les contraintes temporelles justifient une planification soignée.
- *Les routes* : définissent le réseau routier. En métropole, ce réseau est fixe, et donc les routes à emprunter ainsi que les coûts engendrés de leur emprunt sont connus à l’avance. Dans les zones à risques, les routes empruntées sont déterminées en fonction de l’évolution de la crise.
- *Le personnel* : Représente les ressources humaines. Il s’agit de personnel qualifié et obéissant à une hiérarchie.

IV.3.6 Modèle proposé : vers un ordonnancement distribué

Comme évoqué en introduction et contrairement à une approche centralisée de résolution où les décisions d'ordonnancement sont calculées de manière globale, nous proposons de résoudre ce problème par une approche coopérative en considérant que les entités décisionnelles sont décentralisées au sein d'une organisation distribuée. Le problème d'ordonnancement est alors décomposé en plusieurs sous problèmes d'ordonnancement locaux. Chaque agent associé à une zone contrôle sa propre ressource et dispose de sa propre autonomie décisionnelle.

Nous allons dans ce qui suit expliquer le fonctionnement du système au sein duquel des agent-zones vont effectuer l'ordonnancement de leurs tâches de livraison.

IV.3.6.1 Fonctionnement général

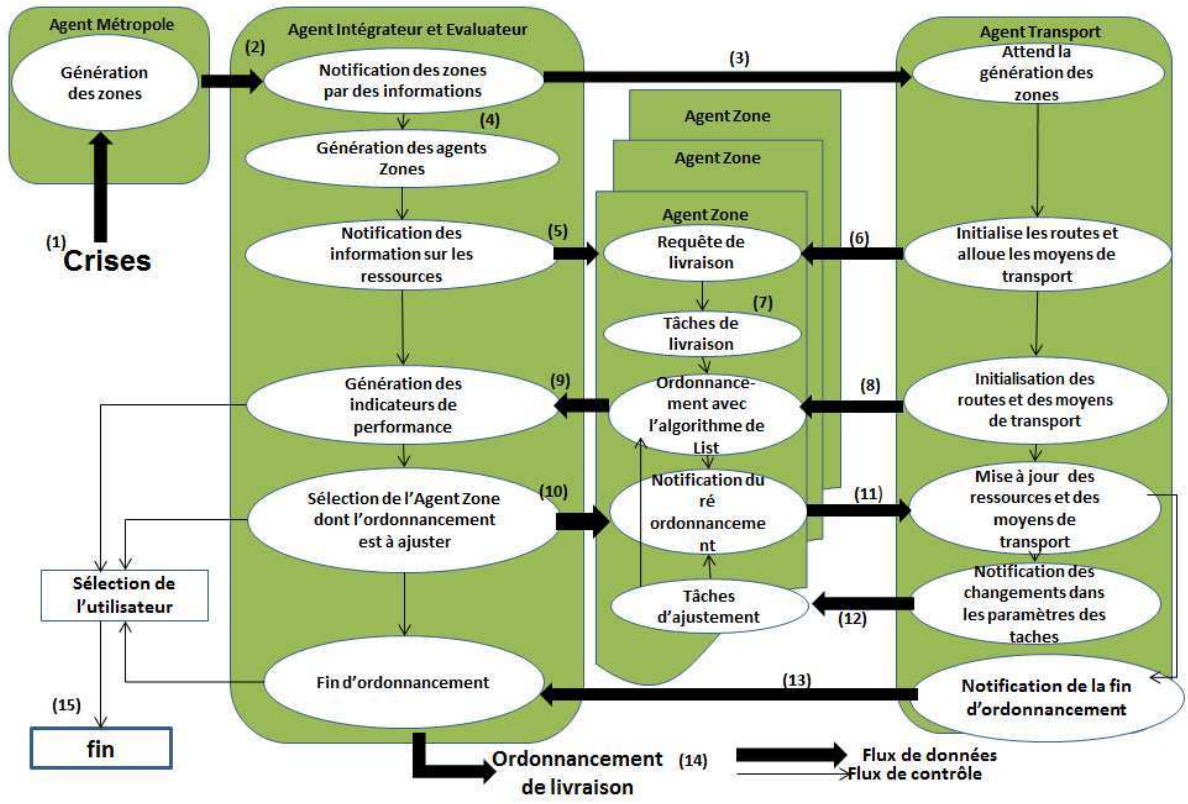

Figure IV. 12 Système d'ordonnancement distribué à base d'agents

a) Lorsqu'une crise est déclarée quelque part dans le monde, l'agent métropole est notifié. Il crée alors les zones de la chaîne logistique de gestion de crise (1). La décision de créer et de positionner les zones est basée sur le module de positionnement optimisé détaillé dans la première section. Les informations liées aux zones générées sont ensuite transmises à l'agent Intégrateur Evaluateur (2).

b) L’agent Intégrateur Evaluateur transmet ensuite ces informations à l’agent Transport (3). Les agent-zones sont ensuite créés afin de gérer chaque zone logistique générée (4) et se voient attribués des informations sur les marchandises à livrer et les quantités en stocks. (5)

c) Chaque agent-zone reçoit un ensemble de demandes de livraisons de la part de ses agents subalternes, qui correspondent aux besoins calculés par l’agent NEA. L’agent-zone « fournisseur » ou « ordonnanceur » transforme ces demandes en tâches de livraison à ordonnancer, en ajustant la valeur de la quantité à livrer selon sa capacité. Avant de procéder au traitement des requêtes de livraison reçues, l’agent zone « fournisseur » sollicite des informations sur les routes et les moyens de transport auprès de l’agent transport, afin de déterminer les valeurs des différents champs qui constituent ses tâches de livraison (d_i et p_i). (6), (7)

d) L’agent Transport utilise les informations précédemment livrées par l’agent Intégrateur et Evaluateur afin d’allouer les routes et les moyens de transport à l’agent-zone pour qu’il puisse commencer son ordonnancement. (8)

e) Chaque agent-zone lance l’ordonnancement local de ses tâches en appliquant un des algorithmes développés dans la couche mathématique de notre architecture. La priorité de ces ordonnancements est le respect de la date limite de livraison. Dès qu’un agent zone obtient une solution, il informe l’agent Intégrateur et Evaluateur du plan de livraison qu’il vient de générer. (9)

f) L’agent Intégrateur et Evaluateur, qui a pour rôle de composer le plan de livraison global à partir des solutions locales reçues, utilise des indicateurs de performance afin d’évaluer l’ordonnancement de la chaîne logistique dans sa globalité. Il détermine ainsi les ordonnancements locaux à améliorer (10). Lorsqu’un plan local est validé, l’agent zone en question le transmet à son supérieur hiérarchique pour qu’il soit pris en compte dans le prochain ordonnancement de ce dernier. Ceci offre l’avantage de garantir une synchronisation entre les agents.

g) Les agent-zones concernés par l’ajustement et le ré-ordonnancement utilisent les indicateurs fournis par l’agent Intégrateur et Evaluateur afin de modifier des paramètres de l’ordonnancement en local (routes, moyens de transport …) et transmettent leurs modifications à l’agent transport. (11)

h) L'agent Transport utilise ces données afin d'allouer les ressources nécessaires pour ces nouveaux ordonnancements et envoie une notification afin d'ajuster les tâches de livraison avec les nouveaux délais de livraison (12). On répète ensuite les étapes 5 à 8, jusqu'à ce qu'il n'y ait plus d'ajustements à effectuer(12). A ce moment-là (13), l'ordonnancement global obtenu est détenu par l'agent Intégrateur et Evaluateur.

i) Si les ajustements ont été une réussite, l'agent Intégrateur et Evaluateur affiche en sortie les résultats de l'ordonnancement. (14) Si l'utilisateur émet une requête pour arrêter le processus, le système s'arrête (15). A la fin du temps de simulation fixé, l'agent Intégrateur et Evaluateur valide la dernière solution optimisée générée.

Les différentes étapes de fonctionnement décrites ci-dessus sont réalisées par les messages ACL échangés entre les agents.

IV.3.6.2 Formulation de notre problème

Nous définissons les variables de notre problème :
La chaîne logistique étudiée a une structure hiérarchique. Ainsi, nous pouvons considérer *A* l'ensemble des agent-zones AgZ_{xy} où *x* est le niveau hiérarchique de l'agent au sein de la chaîne logistique et *y* son rang au niveau *x*.

On a $A = A_T \cup A_I \cup \{AgZ1\}$, avec A_T l'ensemble des agents zones terminales, A_I l'ensemble des agents zones intermédiaires, et *AgZ1* l'agent métropole chargé de créer les autres agent-zones.
Nous avons alors : $A = \{AgZ_1, AgZ2_1, \ldots, AgZ2n_2, AgZ3_1, \ldots, AgZ3n_3, \ldots, AgZm_1, \ldots, AgZm_{ne}\}$, avec *m* le nombre total de niveaux dans la chaîne logistique et n_e le nombre total d'agents au niveau *e*.

Nous notons aussi :

- L'ensemble des ressources à distribuer : $R = \{r_1, r_2, \ldots, r_u\}$, avec *u* le nombre total des ressources ;
- L'ensemble des entrepôts : $S = \{s_1, s_2, \ldots, s_h\}$, avec *h* le nombre total des zones de regroupement de ressources ;
- L'ensemble des moyens de transport : $M = \{m_1, m_2, \ldots, m_k\}$, avec *k* le nombre total des moyens de transport ;
- La capacité de transport du moyen de transport m_j est définie comme étant Cap_{mj} ;

- L'ensemble des routes disponibles : $W_{routes} = \{w_1, w_2, \ldots, w_z\}$, avec z le nombre total des routes ;
- L'ensemble des tâches locales : $T = \{t_1, t_2, \ldots, t_l\}$, avec l le nombre total des tâches à ordonnancer localement ;

Des variables représentant les paramètres des tâches sont également définies:

- c_i = date de fin de la tâche t_i (completion time) ($1 \leq i \leq l$).
- d_i = la date de livraison au plus tard pour la tâche t_i (date de fin souhaitée).
- p_i = durée d'exécution de la tâche t_i.
- *clientID* : Le client ou l'agent-zone qui reçoit la marchandise ou qui attend que la tâche soit effectuée ($clientID \in A$);
- r_j : La ressource à livrer ($1 \leq j \leq u$) ;
- *Qte* : La quantité de ressource à livrer au client ;

Ainsi, une tâche t_i consiste en la livraison d'une quantité (*Qte*) de resource (r_j) à une zone spécifique (agent zone client: *clientID*) en respectant certaines contraintes (la date de livraison souhaitée: d_i et la durée d'exécution: p_i)

$$t_i : < clientID;\ r_j\ ;\ Qte;\ di;\ pi >$$

L'objectif par la suite est de fournir un ordonnancement distribué pour les tâches de livraison exécutées tout au long de la chaîne logistique de gestion de crise.

IV.3.6.3 Construction des tâches de livraison

Pour commencer, chaque agent-zone envoie une demande auprès de son supérieur hiérarchique. L'agent responsable doit déterminer s'il peut répondre aux exigences (étape 5 dans la figure IV.12). Si oui, la requête se transforme en une tâche à accomplir, sinon l'agent-zone responsable calcule la quantité qu'il peut fournir en fonction de son stock et de ses disponibilités, il crée ainsi une nouvelle tâche. Ce calcul est basé sur le degré d'urgence de la requête et sur la date de réception. La procédure « *createTask(v_i)* » (Figure IV.13), avec $v_i \in V$ et V l'ensemble des requêtes, détermine la quantité de ressource à fournir.

Une fois que toutes les requêtes sont traitées, la procédure « *routingTasksConstruction* » (Figure IV.13) donne la liste des tâches prêtes à être exécutées.

Soit V l'ensemble des requêtes et T l'ensemble des tâches ;

```
Procedure routingTasksConstruction
        In:  V
        Out: T
1: begin
2: T= Ø
3: Do
4:      Select v_i Є V
5:              t_i := createTask(v_i)
6:              V := V- {v_i}
7:              T :=  T+ {t_i}
8:While set V of available requests ≠ Ø
9: end
```

Figure IV. 13 Algorithme de création des tâches

La procédure *createTask* transforme une demande de livraison en une tâche à ordonnancer en déterminant la quantité de ressource que l'agent ordonnanceur peut envoyer à chaque agent zone subalternes. Nous expliquons dans la suite le principe de cette procédure :

Lorsque l'agent zone fournisseur ou ordonnanceur reçoit toutes les propositions (les priorités conformes à toutes les ressources voulues) de ses agents subalternes, ou si le temps d'attente Δt_{wait} pour la réception de réponses est expiré, il doit décider comment équilibrer des ressources. Autrement dit, le fournisseur doit examiner quelles sont les ressources à envoyer, à quelles zones et en combien de quantité, selon leurs disponibilités et les priorités des zones. Le fournisseur, va donc évaluer un indexe d'urgence $Eindex\ (r_i, Client_j)$ de chaque ressource r_i ($1 \leq i \leq u$) pour un agent zone client $Client_j$ ($Client_j \in A$):

$$Eindex\ (r_i, Client_j) = Ed\ (r_i, Client_j) \times Ed(Client_j, Ord) \quad (1)$$

Avec

- $Ed\ (r_i, Client_j)$: la priorité de la ressource r_i pour le client $Client_j$,
- $Ed\ (Client_j, Ord)$: la priorité du client $Client_j$ pour l'agent ordonnanceur Ord ($Ord \in A$).

D'après (1), l'index d'urgence d'une ressource r_i pour un agent ordonnanceur Ord (Ord $\in A$) est évalué par:

$$Eindex(r_i,Ord)=Ed(r_i,Ord)\times Ed(Ord,Ord) \quad (2)$$

Avec :

- $Ed\ (r_i, Ord)$: la priorité de la ressource r_i pour l'agent ordonnanceur.
- $Ed\ (Ord, Ord)$: une auto-évaluation de la priorité de l'agent ordonnanceur.

La décision prise par l'agent Ordonnanceur pour partager le stock local restant, dépend des index d'urgence, en sachant que le stock restant pour une ressource r_i pour un agent ordonnanceur est évaluée par :

$$RE_{stock}\ (r_i, Ord)=stock(r_i, Ord)- demand\ (r_i, Ord)\times GEindex\ (r_i, all, Ord)$$

Avec:

- *stock (r_i, AgZ_{xy})* : correspond au niveau local de la quantité de ressource r_i relative à l'agent zone AgZ_{xy}, (pour ce cas particulier : $AgZ_{xy} = Ord$).
- *demand (r_i, AgZ_{xy})*: correspond à la demande de consommation locale de la ressource r_i relative à l'agent zone AgZ_{xy}, (pour ce cas particulier : $AgZ_{xy} = Ord$).
- Sachant que *Ne* correspond au nombre total d'agents incluant ordonnanceurs et clients, *GEindex (r_i, z, AgZ_{xy})* est une pondération de la consommation locale de l'agent-zone identifié par AgZ_{xy} et selon la ressource r_i. Cette expression évalue le degré d'urgence de la consommation de l'agent zone local AgZ_{xy}, par rapport à la consommation totale de tous les agents clients ($z = all$):

$$GEindex\ (r_i, z, AgZ_{xy}) = Eindex\ (r_i , AgZ_{xy}) / \Sigma_{Ne}\ (Eindex\ (r_i , AgZ_{xy}))$$

La prise de décision par un agent ordonnanceur pour partager le stock de sécurité local est finalement évaluée par :

$$\boldsymbol{amount(r_i,Ord,Client_j) = reStock(r_i,Ord) \times GEindex(r_i,others, Client_j)}$$

Une fois la quantité de ressources allouée aux tâches est définie et les tâches sont prêtes (étape 7 dans la figure 16), il ne reste qu'à leur allouer un moyen de transport pour qu'elles s'exécutent.

L'étape suivante consiste à réduire au maximum les retards en ordonnançant l'ensemble des tâches de livraison. Plusieurs algorithmes d'ordonnancement sont applicables, parmi lesquels nous avons choisi l'algorithme de liste, particulièrement adapté à notre chaîne logistique grâce à ses règles de priorité dynamiques, et l'algorithme de *Branch & Bound*. Ces algorithmes sont caractérisés par leur souplesse et la facilité de leur implémentation en temps réel. L'idée dans la suite de mes travaux, est d'enrichir la couche mathématique de la figure I.7 (Cf. Chapitre I) avec d'autres algorithmes d'ordonnancement, en créant ainsi une bibliothèque de méthodes de résolution (Cf. Figure I.7- Chapitre I).

IV.3.6.4 Algorithmes pour l'ordonnancement local des tâches de livraison

Notre approche permet de choisir une méthode d'optimisation parmi celles qui existent dans les bibliothèques pour l'ordonnancement local des tâches de livraisons. Nous présentons à cet effet, l'algorithme de liste et l'algorithme de B&B, que l'agent zone ordonnanceur peut sélectionner en fonction de son environnement, de l'état de son stock, et de la taille de son problème.

a) Algorithme de Liste

Il s'agit d'une approche de résolution possible. Le problème est résolu par des règles de priorité statiques ou dynamiques. L'algorithme de liste sert à arbitrer lorsque plusieurs tâches sont disponibles en même temps. L'algorithme de liste repose sur les principes suivants :

- Principe glouton: ne pas laisser la ressource inoccupée si des taches sont disponibles
- Séquencement glouton des tâches: si la ressource est libre, séquencer la première tâche disponible de la liste
- Arbitrage des tâches: l'algorithme sert à arbitrer avec des règles de priorité lorsque plusieurs taches sont disponibles

Le principe de cette approche consiste à effectuer un ordonnancement des tâches de livraison selon une règle de priorité dynamique basée sur les algorithmes de listes. Plus précisément, à un instant donné T_0, parmi les tâches de livraison disponibles au niveau d'un agent zone, la tâche de plus grande priorité est ordonnancée. Plus généralement, les algorithmes de listes procèdent d'abord à l'élaboration d'une liste de priorités qui est utilisée ensuite pour la construction d'une solution. Dans notre problème la règle de priorité est dynamique. Elle est choisie par l'agent zone responsable de l'ordonnancement. Selon les situations, il peut s'agir de la plus petite date de début au plus tard ou de la plus petite durée d'exécution.

Le principe de notre algorithme est de maintenir une liste pour contenir l'ensemble de tâches prêtes apparues suite à une crise; ces tâches sont une à une exécutées par l'agent-zone qui est responsable de l'ordonnancement. La priorité est donnée à la tâche dont la date de livraison demandée est la plus petite en considérant qu'un retard de livraison peut engendrer des conséquences dramatiques lors d'une crise.

L'objectif étant de minimiser le retard global des livraisons de chaque agent zone, la fonction objectif s'écrit : $Min\ (\sum_{j=1}^{l} max(0, C_j - d_j))$

Algorithme

- Préparer la liste les tâches : Priorité sur les tâches (due date la plus proche) ;
- Séquencement glouton des tâches : quand la ressource (le moyen de transport) est libre, séquencer la première tâche disponible de la liste ;
- Répéter pour toutes les tâches de la liste.

Algorithme de Liste

In: T (Ensemble de tâches à ordonnancer)

Out: S

Début

S={} ;

Tant que T ≠ Ø (ensemble des tâches disponibles T est non vide) **Faire**

Début

Déterminer parmi l'ensemble T, la tâche t_i de plus grande priorité ;

S := S ⊕ t_i

Fin

Fin

Figure IV. 14 Algorithme de liste

Avec :

- *S* est une séquence ou solution partielle (ou totale) de tâches ordonnancées.
- $S = t_i \oplus S$; signifie insérer la tâche t_i dans la séquence *S*, à l'emplacement prévu par l'ordonnancement.

b) Branch & Bound (B&B)

La méthode de Branch and Bound (procédure par évaluation et séparation progressive) consiste à énumérer les solutions d'un problème d'optimisation d'une manière intelligente en ce sens que, en utilisant certaines propriétés du problème en question, cette technique arrive à éliminer des solutions partielles qui ne mènent pas à la solution que l'on recherche.

Par convenance, on représente l'exécution de la méthode de B&B à travers une arborescence. La racine de cette arborescence représente l'ensemble de toutes les solutions du problème considéré.

$$\underset{\alpha_k \in Alpha}{Min} \left(f_{\alpha_k} = \sum_{i=1}^{l} \max(c_i - d_i, 0) \right)$$

Avec: c_i = date de fin d'exécution de la tâche t_i (completion time) ($1 \leq i \leq l$).

d_i = la date de livraison au plus tard pour la tâche t_i.

l = nombre total de tâches à ordonnancer au niveau de chaque agent zone.

Alpha = l'ensemble des nœuds α_k dans l'arbre de recherche de l'agent zone.

L'algorithme1 illustre le fonctionnement global de l'algorithme de B&B pour l'ordonnancement local des tâches.

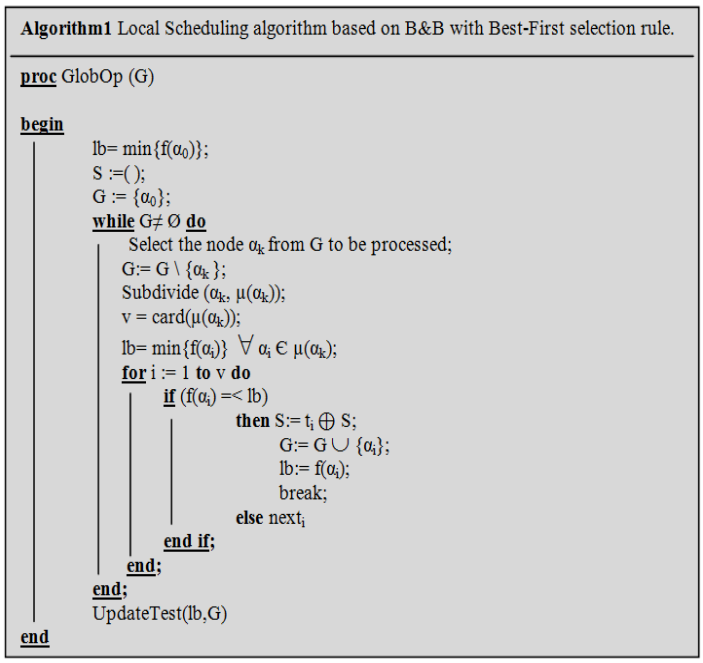

Figure IV. 15 Algorithme de Branch & Bound

Avec :

- α_0 est le nœud racine de l'arbre de recherche *G*.

- *S* est une séquence ou solution partielle (ou totale) de tâches ordonnancées.

- $S = t_i \oplus S$; signifie insérer la tâche t_i dans la séquence *S*, à l'emplacement prévu par l'ordonnancement.

- *lb* est une borne inférieure.

- $\mu(\alpha_k)$ est l'ensemble des nœuds fils de α_k.

- La procedure *Subdivide* $(\alpha_k, \mu(\alpha_k))$ divise le nœud α_k et génère l'ensemble des nœuds fils $\mu(\alpha_k)$.

- $G := G \cup \{\alpha_i\}$ signifie insérer dans l'arbre de recherche G le prochain nœud à explorer.

Pour le problème spécifique que nous abordons ici, une borne inférieure est calculée pour chaque nœud nouvellement créé de la façon suivante:

$$lb = \underset{\alpha_k \in Alpha}{Min} \left(f_{\alpha_k} \right)$$

L'arbre G est alors mis à jour par avec la procédure *updateTest* qui permet à tout moment de supprimer tous les nœuds α_k qui vérifient $f(\alpha_k) > lb$.

Il est intéressant ici de mettre l'accent sur le fait que les agents ordonnanceurs vont pouvoir puiser dans une bibliothèque d'algorithmes d'ordonnancement développée au niveau de la

couche mathématique de notre architecture, afin de choisir l'algorithme le mieux adapté à la situation sur le terrain, en terme de taille de problème et de données (Cf. chapitre I).

IV.3.6.5 Critères ou Indicateurs de performances pour l'évaluation de la chaîne

Cette phase contrôle la validité des solutions en cours, surtout dans le cas de perturbations. Le ré-ordonnancement émerge d'un processus coopératif des agents interagissant pour résoudre la situation troublée et préserver dans la mesure du possible une partie de l'ordonnancement pré-calculé.

Nous avons choisi d'assigner à notre chaîne logistique un coût qui correspond à l'exécution d'un plan d'ordonnancement global retenu. Ce coût est limité par des indicateurs de performance ou critères composés de variables binaires.

Nous définissons les variables suivantes :

Soient p et q deux zones de la chaîne logistique, avec p la zone supérieure hiérarchique directe de q ;

$y_{pq}^{m_j}$: Transport de flux de marchandises de p vers q utilisant le moyen de transport m_j

$$y_{pq}^{m_j} = \begin{cases} 1, & \text{Si le moyen de transport } m_j \text{ est utilisé} \\ 0, & \text{Sinon} \end{cases}$$

$x_{pq}^{w_i}$: Transport de flux de marchandises de p vers q utilisant la route w_i

$$x_{pq}^{w_i} = \begin{cases} 1, & \text{Si la route } w_i \text{ est empruntée} \\ 0, & \text{Sinon} \end{cases}$$

$y_{qp}^{m_j}$: Retour du moyen de transport m_j de q vers p

$$y_{qp}^{m_j} = \begin{cases} 1, & \text{Si le moyen de transport } m_j \text{ est utilisé} \\ 0, & \text{Sinon} \end{cases}$$

$x_{qp}^{w_i}$: Retour en passant par la route w_i

$$x_{qp}^{w_i} = \begin{cases} 1, & \text{Si la route } w_i \text{ est empruntée} \\ 0, & \text{Sinon} \end{cases}$$

On définit ensuite les critères ou indicateurs de performance suivants :

Coût de transport lié aux routes utilisées :

$$C^{\text{routes}} = \sum_{w_i \in W_{routes}} \sum_{p \in A \setminus A_T} \sum_{q \in A_I \cup A_T} (c_{w_i} + c_{pq}) \left(x_{pq}^{w_i} + x_{qp}^{w_i} \right)$$

Coût des camions, avions, bateaux :

$$C^{transport} = \sum_{p \in A \setminus A_T} \sum_{q \in A_I \cup A_T} \sum_{m_j \in M} c_{pq}^{m_j} (y_{pq}^{m_j} + y_{qp}^{m_j})$$

Pénalités en cas de retard :

$$C^{delay} = \sum_{p \in A \setminus \{AgZ_1\}} C_p D_p$$

Coût de stockage en cas d'avance de délivrance :

$$C^{earliness} = \sum_{p \in A \setminus \{AgZ_1\}} \sum_{s_i \in S} c_p^{s_i} E_p$$

Avec :

- c_{w_i} : le coût de l'emprunt de la route w_i ;
- c_{pq} : le coût de chargement à la zone p et de déchargement à la zone q ;
- $c_{pq}^{m_j}$: le coût du moyen de transport utilisé m_j ;
- C_p : le coût de retard de livraison par jour ;
- D_p : le nombre de jours de retard ;
- $c_p^{s_i}$: le coût de stockage dans l'entrepôt S_i de la zone p ;
- E_p : le nombre de jours d'avance de livraison.

Nous mettons en place un ensemble de limites caractérisant les valeurs des critères afin de mieux évaluer la qualité des solutions et des méthodes de résolution. Nous avons alors :

$$C^{routes} = \sum_{w_i \in W_{routes}} \sum_{p \in A \setminus A_T} \sum_{q \in A_I \cup A_T} (c_{w_i} + c_{pq}) \left(x_{pq}^{w_i} + x_{qp}^{w_i} \right) \leq C_{Ref}^{routes} \quad (1)$$

$$C^{transport} = \sum_{p \in A \setminus A_T} \sum_{q \in A_I \cup A_T} \sum_{m_j \in M} c_{pq}^{m_j} (y_{pq}^{m_j} + y_{qp}^{m_j}) \leq C_{Ref}^{transport} \quad (2)$$

$$C^{delay} = \sum_{p \in A \setminus \{AgZ_1\}} C_p D_p \leq C_{Ref}^{delay} \quad (3)$$

$$C^{earliness} = \sum_{p \in A \setminus \{AgZ_1\}} \sum_{s_i \in S} c_p^{s_i} E_p \leq C_{Ref}^{earliness} \quad (4)$$

L'analyse des valeurs des critères permet de déceler les anomalies dans les ordonnancements obtenus en local et d'y remédier. Ces indicateurs nous donnent des informations sur le niveau et l'intensité de la crise au niveau de chaque zone de la chaîne. Seuls les agents zones dont les solutions locales présentent un débordement par rapport aux critères, devront réordonnancer pour améliorer leurs solutions. Un mécanisme de négociation entre agents

zones a été mis en place lors des travaux de Zoghlami (Zoghlami, 2008) pour résoudre les situations bloquantes.

IV.4 Etude de convergence du système proposé

La convergence globale des systèmes multi-agents a fait l'objet de beaucoup d'intérêt dans de nombreux domaines de recherche (Reynolds, 1987 ; Jadbabaie et al., 2003). Dans le but essentiel de contrecarrer les problèmes de complexité exponentielle, nous nous proposons de mettre en place un système de résolution qui se base sur la décomposition du processus global en un ensemble de tâches moins complexes s'exécutant en parallèle. Nous étudions la convergence des agents logistique de notre CLGC pour pouvoir conclure sur la convergence du système global proposé.

IV.4.1 Convergence d'un agent logistique

Afin de prouver la convergence de notre système proposé, nous étudions la convergence des agents logistiques de notre CLGC.

IV.4.1.1 Etude de complexité du comportement de l'agent Métropole

Nous étudions dans cette section, la complexité de l'algorithme de positionnement des zones de l'agent Métropole afin de juger de sa convergence.

La complexité de l'algorithme de positionnement nécessite le calcul de complexité des différentes actions appelées par l'algorithme.
Pour chaque boucle sur l'algorithme principal, on diminue le nombre d'arbres dans la forêt de 1. Et on commence à N. Ainsi lors du $i^{ème}$ passage dans la boucle nous avons :

$$C(i) = c_1 + c_2 + D(i)$$

Avec c_1 et c_2 respectivement les coûts de la première et de la seconde ligne.

D(i) est le coût du calcul des distances.

$$D(i) = (N - i - 2)c_3$$

Où c_3 est le coût le plus défavorable du calcul d'une distance (cas de l'arbre 2-S). En effet, lors du $i^{ème}$ passage, il ne reste que N-i arbres dans la forêt. Lorsque l'on calcule la distance on vient d'en réunifier un de plus. Donc il ne reste que N-i-1 arbres.

Or nous allons simplement calculer la distance entre ce nouvel arbre et les anciens arbres, soit N-i-2 calculs.

Nous avons donc :

$$T(N) = \sum_{i=0}^{N-2} C(i) = \sum_{i=0}^{N-2} c_1 + c_2 + (N - i - 2)c_3$$

$$T(N) = (N-1)(c_1 + c_2) + (N-2)(N-1)c_3 - c_3 \frac{(N-2)(N-1)}{2}$$

$$T(N) = (N-1)(c_1 + c_2) + c_3 \frac{(N-2)(N-1)}{2}$$

$$T(N) \sim c_3 \frac{N^2}{2}$$

En définitive, nous avons prouvé que notre algorithme de positionnement implémenté au cœur de l'agent métropole est polynomial en N^2.

IV.4.1.2 Etude de complexité du comportement de l'agent zone

Le moteur de calcul de l'agent zone est un algorithme d'ordonnancement. Deux voies de résolution des problèmes d'ordonnancement sont considérées : donner une solution exacte selon une stratégie connue, ou bien donner des heuristiques qui permettent d'obtenir des solutions approchées ayant un écart raisonnable par rapport à la solution optimale ou par rapport à une borne inférieur calculée. Selon l'approche choisie, il est facile de démontrer que pour des problèmes de petite taille, l'algorithme converge vers une solution optimale. Les agents zones ordonnanceurs disposent au niveau de la couche mathématique de notre architecture d'une bibliothèque d'algorithmes d'ordonnancements. Selon la situation sur le terrain et la taille du problème, chaque agent ordonnanceur choisit d'appliquer l'algorithme le mieux adapté à son environnement.

Chaque agent zone prendra une décision locale en fonction des capacités de perceptions de l'environnement immédiat. La solution deviendra donc hautement parallèle puisque tous les agents feront leurs calculs en même temps. Le fait de limiter la prise de décisions à une situation purement locale permet également de réduire la complexité de la solution.

IV.4.2 Convergence du système proposé

La principale difficulté induite par l'utilisation de systèmes multi-agents est qu'il est plus difficile de maîtriser le comportement et l'évolution du système global, constitué d'agents autonomes : pour un problème donné, il n'est pas assuré que l'ensemble des agents converge vers une solution commune et unique. Le système d'optimisation des flux proposé s'appuie sur une architecture distribuée composée de plusieurs agents ordonnanceurs qui coopèrent et

interagissent, par échange de messages, suivant une approche décentralisée pour construire un ordonnancement local au niveau de chaque agent, et de manière à converger au mieux vers une solution globale qui satisfait des exigences adoptées collectivement. Nous prouverons dans cette section la convergence de notre solution.

Commençons tout d'abord par définir la notion de "**stratégie**" : Une stratégie est l'ensemble d'actions qui caractérisent le comportement d'un agent dans une société d'agents.

Au départ, et dans une société d'agents, chaque agent se comporte selon une stratégie initiale, mais avec le temps qui passe, il émerge toujours un phénomène global que la plupart des agents tendent à suivre, convergeant ainsi vers une stratégie comportementale, que l'on appelle la synchronisation collective.

Nous nous intéressons dans ce contexte à la notion d'émergence et ses implications dans un SMA.

IV.4.2.1 Caractérisation d'un phénomène émergent

En éthologie et en entomologie, un comportement émergent est un effet global qui résulte de l'application de règles locales, généralement très simples. Par exemple, il est possible de simuler l'activité très complexe d'une fourmilière à l'aide de règles très simples concernant chaque fourmi (Drogoul, 1993). En science des systèmes, l'accent est mis sur le phénomène d'émergence. Ce n'est plus une somme de comportements simples mais le résultat d'une interaction entre ces comportements et la complexité du système. Nous proposerons donc une approche distribuée à ce problème qui s'appuie sur la coopération d'agents réactifs permettant d'obtenir la solution par émergence du comportement global de la CLGC.

Un récent travail de bibliographie sur la notion d'émergence, en particulier dans les SMA, est fourni par (Deguet, 2008). Ce travail est axe sur ce qui caractérise un phénomène émergent et conclut sur l'importance primordiale des interactions par rapport aux comportements individuels pour leur apparition. Il définit l'émergence comme un « avis plus ou moins consensuel, sans fondement formel, regroupant tout ou partie des critères » :

• Causalité descendante : l'émergent, ou épiphénomène, est perçu indépendamment des entités qui composent le système, même s'il provient d'elles, et a une causalité de son niveau global sur le niveau local de ces entités.

• Observation et prédiction : l'émergence facilite la description de l'état du système ou de son évolution future. Mais il n'y a pas de réduction possible : on ne peut pas déduire le niveau global du niveau local. L'observation du système est nécessaire, et le meilleur moyen de prévoir son évolution est de le simuler.

• Complexité à partir de la simplicité : l'émergence est une description simple d'un comportement complexe.
• Interprétation : l'émergence est l'interprétation d'un phénomène résultant d'une interaction entre les agents et l'environnement. L'observation et l'interprétation peuvent être faites par une personne extérieure au système (émergence faible) ou par les entités locales elles-mêmes (émergence forte).
• Emergence et auto-organisation : bien qu'elles se rencontrent souvent ensemble, ces notions ne font pas consensus. Pour (DeWolf, 2005), elles sont indépendantes mais toutes deux positives, et c'est leur combinaison qui fait leur force. Pour (Shalizi, 2001), au contraire, l'auto-organisation génère la complexité du système, ce qui incite à rechercher des régularités émergentes pour la réduire.
• Le tout est supérieur à la somme des parties : cette phrase est classique de la caractérisation de l'émergence, et traduit le gain apporte au système par l'interaction de ses composants locaux.

L'une des questions principales liées à l'émergence et aux SMA est de déterminer comment *concevoir le niveau local* du système (agents, interactions et organisation) pour garantir l'obtention d'un comportement émergent particulier au niveau global.

La construction d'un SMA vise à la *réalisation d'objectifs globaux* en définissant son niveau local. Son évolution dépend ensuite de données fournies par l'environnement, d'une initialisation particulière, ou de la modification d'agents due à l'ouverture du système. Nous prouverons dans la suite que le phénomène global qui émerge dans notre SMA converge vers une stratégie dominante.

IV.4.2.2 Convergence proéminente de notre système

Nous nous inspirons dans la suite des travaux de (Jiang, 2008) pour prouver la convergence de notre système.

Dans les sociétés d'agents, nous pouvons toujours apercevoir un phénomène intéressant : s'il existe des agents dont les rangs sociaux sont plus élevés que ceux d'autres agents ordinaires, ou dont les stratégies sont plus dominantes que celles des autres agents ordinaires (les agents ayant des rangs dominants ou stratégies dominantes sont appelés agents de premier plan), alors la synchronisation collective finit par converger vers les stratégies de ces agents de premier plan : c'est ce qu'on appelle la convergence proéminente.

Soit S l'ensemble de toutes les stratégies s_i de notre système. Pour des raisons de simplicité, nous affectons un rang (entier naturel) à chaque stratégie s_i dans l'espace des stratégies S et

nous supposons que plus la valeur d'une stratégie est grande, plus la stratégie est dominante dans la synchronisation collective.

Dans notre système, chaque agent est doté d'un rang social. Les agents des différentes classes sociales vont alors avoir différents effets sur le système : les agents supérieurs peuvent facilement influencer les stratégies des agents subalternes.

Définition 1. Le rang social de l'agent ai peut être exprimé par la fonction: $p_i \rightarrow R$.

$p_i \geq p_j$ signifie que a_i est de rang supérieur à a_j.

L'ensemble des rangs sociaux de tous les agents dans le système peut être exprimé par: P = $[p_i]$, où $1 \leq i \leq n$, et *n* désigne le nombre d'agents dans le système.

Définition 2. (Capacité de synchronisation des agents individuels).

De toute évidence, plus le rang social de l'agent est élevé alors plus celui-ci est capable de participer à la synchronisation collective; d'autre part, selon notre hypothèse, plus la valeur de la stratégie est grande, plus celle-ci est dominante dans la synchronisation collective.

Par conséquent, la capacité de synchronisation d'un agent est déterminée par son rang social et sa stratégie.

Soit s_i la stratégie de l'agent a_i, p_i désigne le rang social de a_i, A désigne l'ensemble des agents sur le terrain, la capacité de synchronisation c_i de a_i, peut être définie comme:

$$c_i = p_i . \frac{s_i}{\frac{1}{|A|}\sum_{j \in A} s_j}$$

où | A | désigne le nombre total de tous les agents dans le champ de synchronisation.

La synchronisation collective de notre système multi-agents suit toujours plusieurs hypothèses:

- Le résultat de la synchronisation est la convergence vers une stratégie commune émergente: Les différentes stratégies individuelles des agents vont toujours converger vers une stratégie commune du système, qui est dans notre cas l'obtention d'un calendrier global acceptable de la CLGC;
- Au départ, les effets de tous les agents sur la synchronisation sont identiques et tous les agents ont la même capacité de synchronisation;
- Tous les agents zones ordonnanceurs peuvent faire appel aux algorithmes d'optimisation implémentés au niveau de la couche mathématique. Pour chacun, les données d'entrée et sortie dépendent de sa propre perception de son environnement et de la stratégie choisie.

Analysons maintenant la convergence de la stratégie de chacun des agents de notre système.

- La stratégie de l’agent Intégrateur Evaluateur est d’évaluer la validité des solutions locales en se basant sur des indicateurs de performances prédéfinis, et d’envoyer des notifications de ré-ordonnancement aux agents zones ordonnanceurs. Des conditions suffisantes sont fournies, qui garantissent la convergence de cet agent ordinaire.

- La stratégie de l’agent Transport est d’allouer les moyens de transports et les routes praticables pour chaque agent zone ordonnanceur, en vue de l’acheminement des ressources. Des conditions suffisantes sont fournies, qui garantissent la convergence de cet agent ordinaire.

- La stratégie de l’agent Métropole consiste en la création et le déploiement de la chaine logistique de gestion de crise, en se basant sur l’arbre de Steiner pour minimiser le coût de la chaîne créée. La convergence de cet agent de premier plan a été prouvée dans la section IV.3.2 (IV.3.2.1Convergence d’un agent logistique).

- La stratégie de chacun des agents zone ordonnanceurs, qui sont considérés dans notre cas comme des agents de premier plan, est de fournir un ordonnancement local optimisé pour l’acheminement des ressources vers les zones subalternes, en utilisant des algorithmes d’optimisation connus à l’avance. Chaque agent zone ordonnanceur peut puiser dans la couche mathématique de notre architecture pour choisir l’algorithme le mieux adapté à sa situation. Grace à l’aspect distribué de notre approche, les données d'entrée et de sortie de chaque sous problème d’ordonnancement sont en quantités limitées et les différentes tâches à planifier sont en nombre limité. Chaque agent zone est confronté généralement à un problème d'ordonnancement de petite taille. Cela devrait économiser considérablement le temps de calcul et garantir la convergence de chacun de nos agents-**zones de premier plan**, comme prouvé dans la section IV.3.2 (IV.3.2 Convergence d’un agent logistique)

En se référant aux définitions présentées dans les travaux de Jiang (Jiang, 2008), nous avons identifié les agents de premier plan et les agents ordinaires de notre CLGC. En partant de l’hypothèse que les agents de même types ont tendance à suivre la même stratégie, et que les agents ayant un rang social plus élevé possèdent un degré de convergence dominant par rapport aux autres agents, et en se référant à la section précédente, nous prouvons que les stratégies des agents ordinaires ont tendance à converger vers les stratégies dominantes des agents de premier plan. Nous obtenons donc l’émergence du phénomène de convergence globale proéminente dans notre système.

IV.5 Conclusion

Dans ce chapitre, nous avons présenté deux approches d'optimisation pour la mise en œuvre d'un système d'aide à la décision au service de la logistique militaire. Les approches adoptées sont intégrées dans un système d'information multi-agent dont le comportement vise à satisfaire les consommateurs en zone de crise, en termes du temps d'attente des livraisons, et satisfaire les gestionnaires du système, en termes du cout de gestion de la chaîne logistique.

L'approche d'optimisation réactive proposée se base sur une interaction dynamique et flexible entre les agents zones et des agents connexes pour se coordonner efficacement afin de couvrir l'intégralité des tâches à traiter lors d'une crise. Nous avons donc présenté dans ce chapitre les processus d'optimisation des flux adoptés.

Nous présentons dans le chapitre suivant, la plateforme de développement multi-agent utilisée, avant d'illustrer la pertinence de nos approches par des scénarii de simulations et les résultats obtenus.

Chapitre V : Outil de simulation proposé pour différentes situations logistiques

V.1 Introduction

Nous avons développé un système multi agent aussi général que possible qui traite du problème de la gestion des flux en cas de crise. Ce système a pour but de vérifier l'efficacité de notre approche en termes de temps de stabilisation de la crise, de qualité des solutions et de respect des contraintes. Dans ce dernier chapitre, nous détaillons les différents résultats issus de l'optimisation. En conséquence, nous présentons dans un premier temps la plate-forme de simulation choisie ainsi que les outils utilisés. Par la suite, nous progressons dans les simulations en faisant un suivi de l'état de crise suivant la modélisation adoptée de la CLGC.

V.2 Les plateformes de développement

Le besoin de mettre en œuvre des systèmes à plusieurs composantes autonomes nécessite une infrastructure de logiciels utilisée comme environnement pour le déploiement et l'exécution d'un ensemble d'agents. Cette infrastructure est appelée ***plate-forme de développement des systèmes multi-agents***. Une plate-forme de développement des systèmes multi-agents est une infrastructure de logiciels utilisée comme environnement pour le déploiement et l'exécution d'un ensemble d'agents.

Cependant, l'implémentation de tels systèmes s'avère souvent difficile au niveau de la manipulation de structures de données complexes, de la distribution, de la communication ainsi qu'au niveau des contraintes matérielles imposées. En plus, l'intelligence artificielle est un domaine de recherche extrêmement riche et cette richesse induit une grande complexité et une grande multiplicité des approches proposées, ce qui conduit à de très nombreux modèles d'agents, d'environnement, d'interactions et d'organisations. Ces modèles sont souvent combinés au sein d'un même système multi-agent. Ainsi, le mieux est de choisir une plateforme multi-agent adaptée aux contraintes du système à mettre en œuvre. En recherchant dans les outils disponibles, nous pouvons trouver une variété de plateformes qui permettent la mise en place et le suivi du processus de développement à base d'agents. Comme le choix d'une plateforme d'agent a une grande influence sur la conception et la mise en œuvrer des agents, FIPA (Fédération International Agent) a produit les normes qui concernent comment une plateforme d'agent devrait être (ACL). Ces normes existent pour assurer une conception uniforme des agents indépendamment de la plateforme. La plupart de ces outils sont

développés avec Java. Nous donnons dans ce qui suit une brève présentation des outils les plus cités et utilisés.

V.2.1 CORMAS :

Abréviation de *COmmon Resources Multi-Agent System*, CORMAS est une plateforme libre (open source) basée sur SmallTalk comme langage de programmation pour le développement de systèmes multi-agent. Spatialisée, cette plateforme est plutôt orientée vers des problématiques de recherche en sciences du développement et de négociation entre acteurs.

V.2.2 Zeus

Il s'agit d'une plateforme logicielle (Nwana et al., 1999) développée par British Telecom System Research Lab où des systèmes collaboratifs peuvent être élaborés en quatre phases : analyse, conception, réalisation et support à l'exécution. Elle définit les rôles au début de l'élaboration, pour ensuite décrire l'organisation et enfin la coordination. Elle présente par ailleurs l'inconvénient majeur d'être très complexe et difficile de maitrise et nécessite donc de longs apprentissages.

V.2.3 MadKit

Madkit4 (Multi-Agent Developpment Kit) créé en 1996 par Olivier GUTKNECHT et Michel FERBER (Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier). MadKit est une plateforme multi-agent entièrement écrite en Java. Elle se base sur un modèle organisationnel (Agent/Groupe/Rôle).

La communication dans Madkit est basée sur un mécanisme de peer to peer, et permet le développement rapide des applications distribuées en utilisant les principes multi-agent. Parmi les outils que Madkit propose, on trouve une interface graphique qui permet de supporter différents modes d'utilisation et d'exploitation de la plateforme.

V.2.4 Jade

Acronyme de *Java Agent DEvelopment*, *JADE* est un framework basé sur le langage *Java* et a été développé à l'Université de Parme en Italie. C'est une plateforme JAVA initiée pour le développement des SMA et répond aux normes *FIPA*. A ce stade, il convient tout d'abord de définir la norme *FIPA*50 (Foundation for Intelligent Physical Agents) avant de procéder à la description de *JADE*.

- *FIPA (Fondation for Intellegent Physical Agents)*

Etant donnée la diversité remarquable des travaux de recherche touchant au domaine du multi-agent, une normalisation s'imposait et la *FIPA* a ainsi été initiée, en 1996, comme projet

de normalisation du concept agent. Ce projet a été mis en place, et plein d'autres encore, en vue d'instaurer une interopérabilité tangible entre agents et concepteurs. A signaler dans ce contexte, les projets *AUML*51 (*Agent UML*) et *FIPA* à laquelle nous nous intéressons particulièrement. Celle-ci promet en effet une meilleure interaction et communication inter agents via des moyens standardisés, sans pour autant porter atteinte à leur sens initial.
Les documents FIPA décrivent le modèle de référence d'une plate-forme multi-agents ou ils identifient les rôles de quelques agents clés nécessaires pour la gestion de la plateforme, et spécifient le contenu du langage de gestion des agents et l'ontologie du langage. Trois rôles (agents) principaux sont identifiés dans une plate-forme d'agent :
– Le DF "Director Facilitor" est un module qui fournit un service de pages jaunes à la plateforme.
– Le ACC "Agent Communication Channel" gère les différentes communications entre les agents.
– Le AMS "Agent Management System" supervise l'enregistrement des agents, leur authentification, leur accès et l'utilisation du système.

Jade utilise pour la communication entre agents le langage FIPA ACL. Il s'agit d'un langage standard et impose le codage, la sémantique et la pragmatique des messages.

V.3 Choix de la plateforme

La portabilité pourrait être assurée en choisissant une plateforme développée en langage Java. En effet, le langage Java est portable ce qui facilite la portabilité de la plateforme. Or la plupart des plateformes multi-agent sont à base de java, la portabilité est donc assurée dans la majorité des cas.
Pour la sélection de la plateforme, nous avons négligé les critères insignifiants telles que la difficulté d'apprentissage ou la non disponibilité des sources. Néanmoins, nous avons souligné quelques critères importants :

- La possibilité d'implémenter des systèmes relativement complexes ;
- La flexibilité : éviter les plates-formes qui supportent une méthodologie particulière ;
- L'accélération de développement grâce à la présence suffisamment importante de briques logicielles pour pouvoir produire une application aboutie ;
- Le traitement distribué et notamment la présence d'un support pour le paradigme AM ;
- Possibilité d'intégration des services web.

- La portabilité.

Les deux plateformes qui ne spécifient aucune méthodologie et peuvent être considérées comme des « frameworks[2] », sont JADE[3] et Jack mais JADE l'emporte avec plusieurs autres caractéristiques intéressantes telles que la possibilité d'intégration des Web services et l'existence d'un bon support de langages de contenu et d'ontologies.

Ainsi, pour le développement de notre système et la simulation des résultats de nos approches d'optimisation distribuée, nous avons choisi la plateforme JADE (Java Agent Development framework). JADE est un logiciel-médiateur [4]"middleware" qui permet une implémentation flexible des Systèmes Multi-Agents communiquant grâce à un transfert efficace des messages ACL (Agent Communication Language), conformes aux spécifications de la FIPA. JADE est écrit en Java, supporte la mobilité, évolue rapidement et fait partie aujourd'hui des rares plateformes multi-agents qui offrent la possibilité d'intégration des services Web (Green, 2005. D'un autre côté, JADE tente de faciliter le développement des applications agent en optimisant les performances d'un système d'agents distribué.

V.4 Outils et paramétrages

V.4.1 Plateforme et conteneur

Chaque instance de JADE s'exécutant simultanément sur un ou plusieurs postes est appelée conteneur (container). Un ensemble de conteneurs actifs est appelé plateforme. Pour chaque plateforme il existe toujours un et un seul conteneur principal appelé main container. Ce conteneur héberge l'AMS Agent Management System) et DF (Directory Facilitator).Tous les autres conteneurs doivent se déclarer dans le « main container ».

[2] Un ensemble de bibliothèques permettant le développement rapide d'applications. Il fournit suffisamment de briques logicielles pour pouvoir produire une application
[3] http://jade.tilab.com
[4] Permet la communication entre des clients et des serveurs ayant des structures et une implémentation différentes

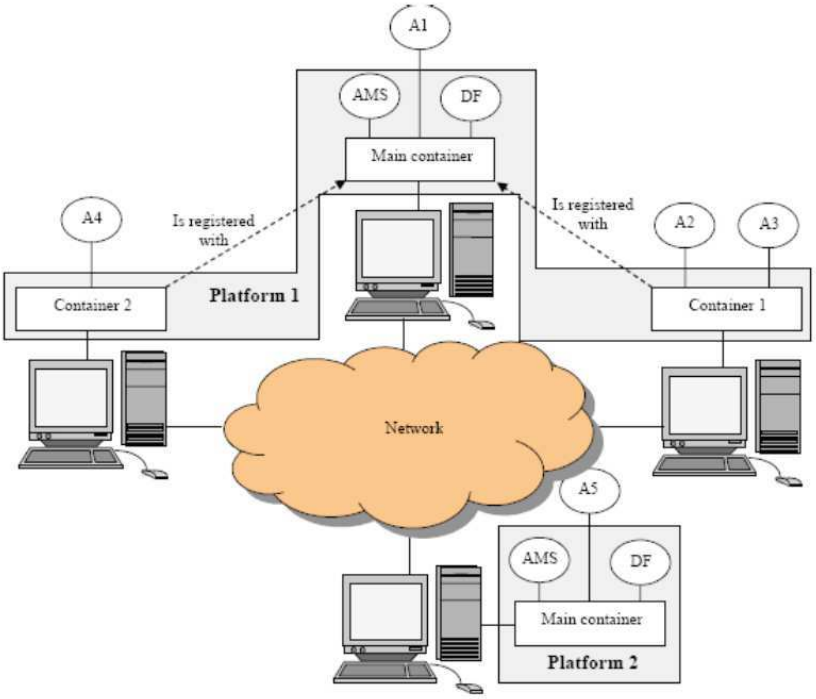

Figure V. 1 Plateformes et Conteneurs

V.4.2 Outils utiles au débogage

La plateforme JADE offre des outils utiles au débogage :

- *Dummy Agent*
    - Visualisation des messages
    - Envoie des messages aux agents présents sur la plateforme et réceptionne leur réponse,
- *Sniffer Agent*
    - Visualisation de l'enchaînement des messages, et des messages eux même.
    - Vérification interactive de la correction des protocoles.
- *Introspector Agent*
    - Visualisation des messages envoyés/reçus,
    - Visualisation des Behaviours actifs et non actifs,
    - Contrôle de l'état de l'agent.

V.4.3 Langage ACL (Agent Communication Langage)

Les ACL prennent place dans une couche logiquement supérieure à celle des protocoles de transfert (TCP/IP, HTTP, IIOP) et adresse le niveau intentionnel et social des agents. Les deux ACL les plus connus sont KQML (*Knowledge Query and Manipulation Language)* et FIPA-ACL. Nous allons particulièrement nous intéresser à l'ACL de la FIPA.

Les agents communiquent entre eux en échangeant ce qui peut représenter des actes de langage, et qui sont encoder dans un agent de langage de communication ACL. Nous avons listé quelques-unes des communications possibles les plus utilisées et les avons regroupées selon les actions suivantes :

Actions	Syntaxe	Définition - Sens
Accept Proposal	accept-proposal	Communication de l'accord de l'expéditeur d'effectuer une action qui lui a été préalablement soumise.
Agree	agree	Communication de l'accord de l'expéditeur pour effectuer une action, sans doute dans le futur.
Cancel	cancel	Communication de l'annulation de l'accord donnée préalablement par l'expéditeur pour effectuer une action.
Call for Proposal	cfp	Communication par l'expéditeur d'une demande d'effectuer une certaine action.
Confirm	confirm	Communication par l'expéditeur de la confirmation de la validité (selon les règles de l'agent) de la proposition préalablement reçue.
Disconfirm	disconfirm	Communication par l'expéditeur de la confirmation de la non validité (selon les règles de l'agent) de la proposition préalablement reçue.
Inform	inform	Communication par l'expéditeur d'une proposition, pensée vrai par celui-ci.
Propose	propose	Communication par l'expéditeur d'une proposition d'action conditionnée à certaines préconditions données.
Request	request	Communication par l'expéditeur d'une demande au destinataire d'effectuer une action.

Tableau V. 1 Quelques actions de communications

Le message type du Fipa ACL contient tout d'abord le type du message envoyé (= syntaxe de ce message), mais aussi :

- l'expéditeur du message
- le destinataire du message
- le contenu du message
- le langage utilisé dans le contenu du message ("language ..."),
- le protocole utilisé,
- l'ontologie auquel le message se rattache ("ontology ..."),
- la référence d'un message antérieur auquel le message actuel se rattache ("in-reply-to ..."), ou la référence d'un message ultérieur attendu en retour ("reply-with ...").
- la référence de la conversation.

V.5 Simulations et Résultats

Cette application a pour but de gérer les flux de marchandises et l'évolution des stocks d'une chaîne logistique militaire. Lors d'une opération militaire, une chaîne logistique se met en

place avec une base logistique de ZRR. D'autres zones logistiques en découlent pour relier la base logistique ZRR à la zone d'opération.

Nous commençons par présenter notre démonstrateur OBAC de manière générale, ensuite nous détaillons les résultats obtenus du module de positionnement des zones logistiques, à travers plusieurs scénarii de simulations.

Par la suite nous présentons les résultats de simulations du module d'estimation des besoins, en les comparants avec les résultats obtenus avec la première version du NEA.

Finalement, l'impact des comportements des agents zones sur l'évolution des stocks et l'optimisation des flux sera présenté lors de la dernière section.

V.5.1 Présentation du Démonstrateur OBAC

Au démarrage de l'application, une première fenêtre de configuration apparaît. Celle-ci permet de choisir le nombre de base logistique Terminales (ZT) que l'on souhaite associer à la ZRR, ainsi que le nombre de ressources que l'on souhaite représenter pour la simulation.

On choisit aussi la durée souhaitée pour la simulation [de 1 à 365 jours].

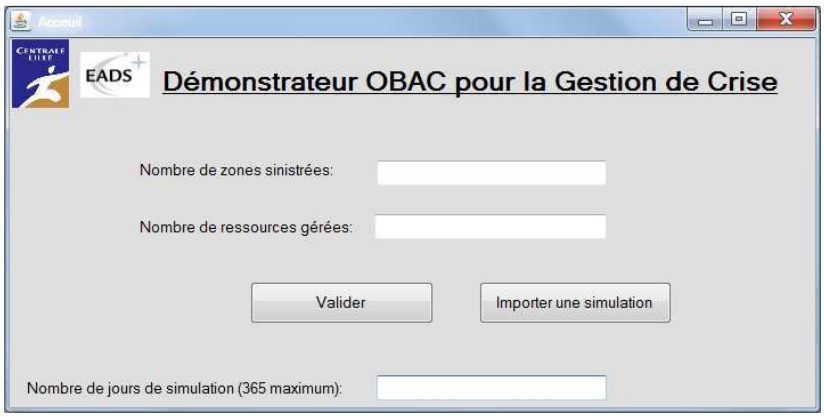

Figure V. 2 Interface initiale du démonstrateur OBAC

Une fois cette première fenêtre validée, une seconde fenêtre de configuration s'affiche à l'écran. Cette fenêtre permet à l'utilisateur de renommer les ressources, mais surtout de définir la quantité moyenne (en unité de ressource) de ressource consommée par un soldat, en temps de repos (non combat). Ces valeurs serviront de base pour l'estimation des besoins des zones. Toutefois, l'exactitude de ces valeurs n'est pas indispensable dans la mesure où l'estimateur de besoins convergera de toute façon vers les valeurs réelles.

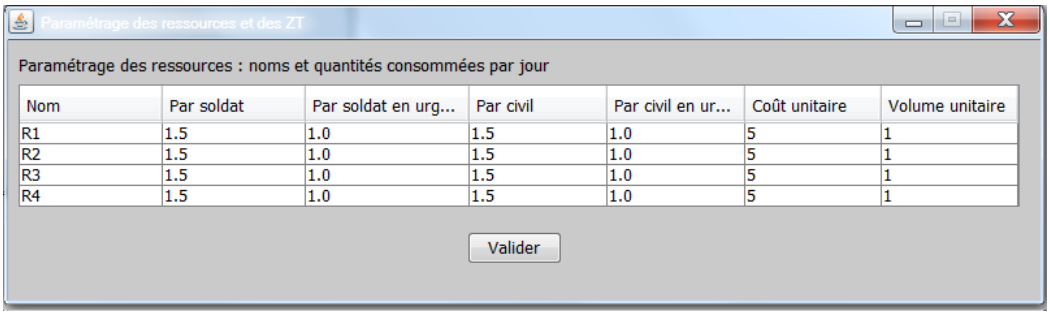

Figure V. 3 Paramétrage des ressources

Une fois cette deuxième fenêtre validée, la fenêtre principale du démonstrateur OBAC s'ouvre (Figure V.4).

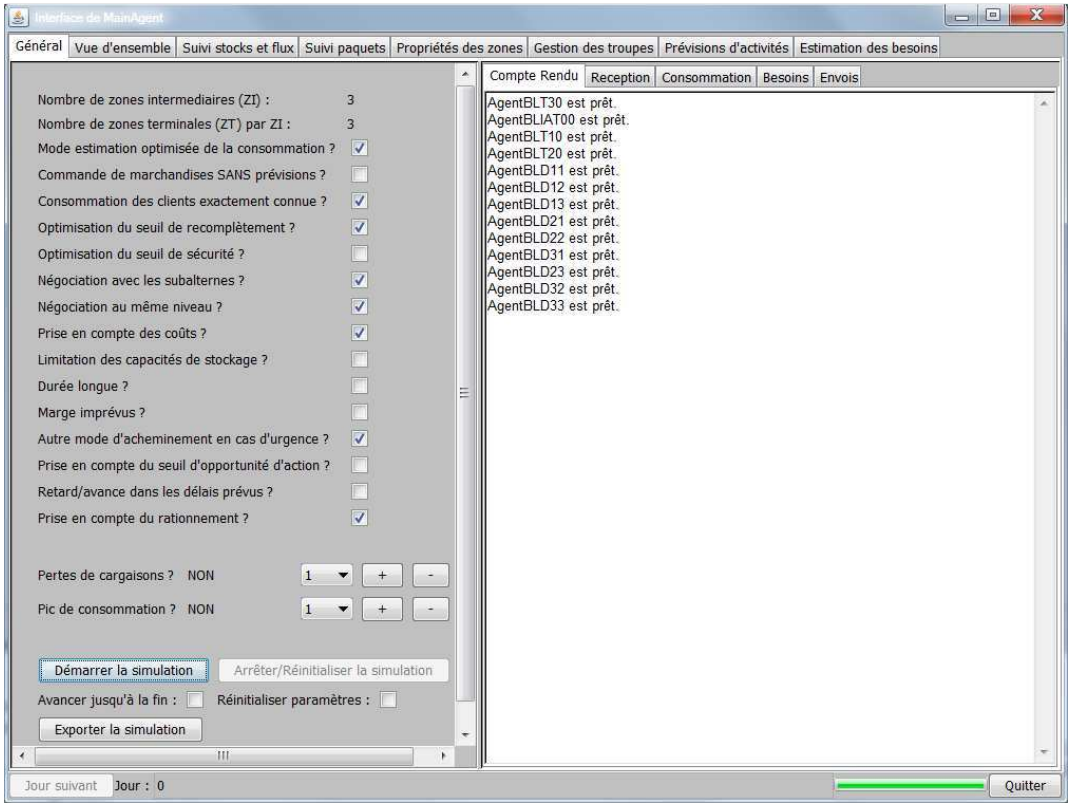

Figure V. 4 Interface graphique principale

Cette fenêtre permet de définir différents modes pour la simulation.

Avant de démarrer la simulation, des modifications éventuelles sont à réaliser sur quatre des onglets de l'application.

1. L'utilisateur choisit les modes de simulation qu'il souhaite dans l'onglet « Général ». Il a également la possibilité d'imposer des perturbations dans ce même onglet ;
2. Une fois les modes de simulation choisis, l'onglet « propriétés des agents » permet de modifier les valeurs internes de chacune des zones logistiques ;

3. Si des mouvements de troupes sont prévus, l'onglet « Gestion des troupes » permet d'enregistrer les mouvements prévus entre les différentes zones logistiques ;
4. Enfin, si des opérations particulières sont prévues sur n'importe laquelle des zones logistiques, l'utilisateur a la possibilité de renseigner l'application sur les détails de l'opération, afin que l'estimateur de besoins puisse en tenir compte et anticiper les changements de consommations dus à des changements d'activités.
5. Une fois les quatre points précédents renseignés, le bouton « Démarrer la simulation » de l'onglet « Général » permet de lancer la simulation.

L'onglet « Vue d'ensemble » (Figure V.5) permet d'avoir une vue sur la structure hiérarchique de la CLGC, avec les niveaux de stocks en chaque ressources et au niveau de chacune des zones.

L'onglet Suivi des paquets permet de visualiser les marchandises en cours de livraison dans, où les colis en retard (ou perdus) s'affichent en rouge. La coloration des lignes est fonction des zones de départ et d'arrivée :

- Gris entre MainAgent et la ZRR
- Violet entre la ZRR et les ZI,
- Vert entre les ZI et les ZT,
- Rouge entre deux ZT.

Expéditeur	Destinataire	Type de ressource	Quantité	Date d'envoi	Date de réception	N° du colis
MainAgent	AgentBLIAT00	vivres	903	1	3	0
MainAgent	AgentBLIAT00	R2	903	1	3	2
AgentBLIAT00	AgentBLT20	vivres	176	1	4	4
AgentBLIAT00	AgentBLT20	vivres	60	1	3	5
AgentBLIAT00	AgentBLT20	R2	176	1	4	6
AgentBLIAT00	AgentBLT20	R2	60	1	3	7
AgentBLT20	AgentBLD23	vivres	225	1	6	8
AgentBLT20	AgentBLD23	R2	225	1	6	9
AgentBLIAT00	AgentBLT10	vivres	176	1	4	10
AgentBLIAT00	AgentBLT10	vivres	60	1	3	11
AgentBLIAT00	AgentBLT10	R2	176	1	4	12
AgentBLIAT00	AgentBLT10	R2	60	1	3	13
AgentBLT10	AgentBLD11	vivres	225	1	6	14
AgentBLT10	AgentBLD11	R2	225	1	6	15
AgentBLT20	AgentBLD21	vivres	225	1	6	16
AgentBLT20	AgentBLD21	R2	225	1	6	17
AgentBLIAT00	AgentBLT30	vivres	176	1	4	18
AgentBLIAT00	AgentBLT30	vivres	60	1	3	19
AgentBLIAT00	AgentBLT30	R2	176	1	4	20
AgentBLIAT00	AgentBLT30	R2	60	1	3	21
AgentBLT30	AgentBLD31	vivres	225	1	6	22
AgentBLT30	AgentBLD31	R2	225	1	6	23
AgentBLT10	AgentBLD13	vivres	225	1	6	24
AgentBLT10	AgentBLD13	R2	225	1	6	25

Figure V. 5 Suivi des paquets

L'onglet Suivi stocks et flux permet d'afficher l'évolution des stocks des différents agents pour les différentes ressources. Notamment, les niveaux de stock des jours passés sont affichés en bleu, et les niveaux à venir sont en bleu ciel et beige.

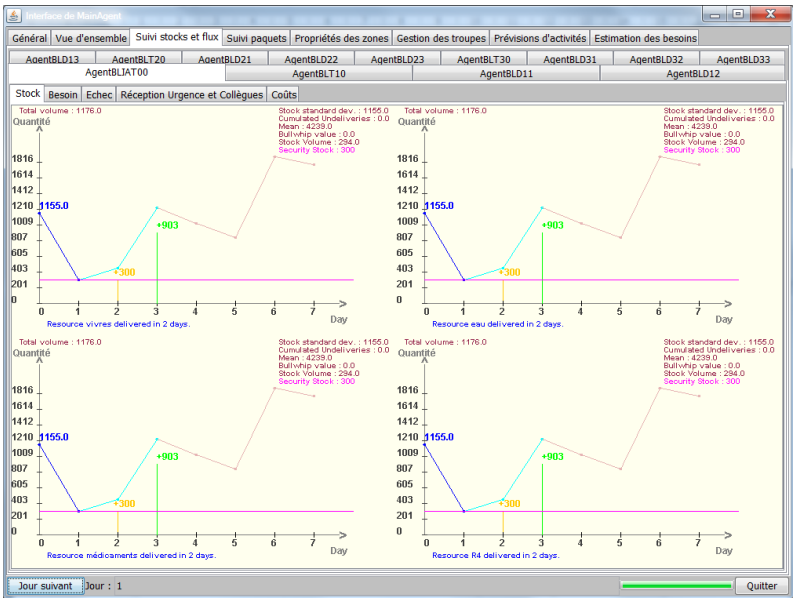

Figure V. 6 Evolution des stocks

V.5.2 Positionnement des zones : Le mode automatique

Dans cette section nous présentons les résultats de nos tests sur l'algorithme de positionnement des zones logistiques que nous avons implémenté. Nous avons pour cela construit différents scénarii recouvrant la quasi-totalité des cas possibles. Lors de ces tests, l'utilisateur rentre les coordonnées de zones connues au départ, à savoir la zone métropole et les zones sinistrées. Ces zones seront appelées ***« zones utilisateurs »*** pour les besoins de la simulation et elles seront représentées en **bleu** sur la carte et libellées d'un ***U*.**

L'algorithme crée alors des zones dite ***« intermédiaires »*** qui seront représentées en **rouge** sur la carte et libellées d'un ***I*.**

Lorsque les zones utilisateurs sont réparties sur des continents différents, l'outil demande à l'utilisateur (à l'aide d'une nouvelle fenêtre) de placer sur la carte de nouvelles zones. Ces zones sont appelées ***« zones portuaires »***. Elles sont représentées sur la carte de la même façon que les zones utilisateurs : couleur bleue et lettre U.

Les résultats des tests sont présentés de la manière suivante :

- Un tableau récapitulant les données entrées par l'utilisateur;
- L'interface de saisie de données concernant les coordonnées des zones utilisateurs ;
- Un imprimé de l'écran contenant les coordonnées des éventuelles zones portuaires ;
- L'interface graphique de la carte du monde où figurent toutes les zones ;
- Les résultats retournés par l'algorithme dans la console à savoir : le poids de l'arbre généré (équivalent de la distance cumulée entre les zones) et les coordonnées géographiques des zones intermédiaires créées.

V.5.2.1 Premier test : cas de 2 zones utilisateurs

Les tests du cas à deux zones sont des plus triviaux. En effet, dans ce cas de figure, l'outil ne génère pas de zones intermédiaires en considérant qu'il n y a pas d'optimisation à faire en terme de distance pour relier les deux zones (le plus court chemin entre deux points est la ligne droite). Cependant, la seconde sous partie montre un exemple avec la création de zones portuaires, dans le cadre de la prise en compte de la « contrainte eau » (nous avons fait en sorte qu'aucune zone intermédiaire ne soit placée en plein milieu d'un océan).

a) Sur un même continent

Données de départ :

Zones initiales	Latitude	Longitude
Tokyo	35.69	139.69

Zones terminales	Latitude	Longitude
Fukushima	37.76	140.47

Zones portuaires	Latitude	Longitude
-	-	-

Tableau V. 2 Données de départ pour le cas 2 zones

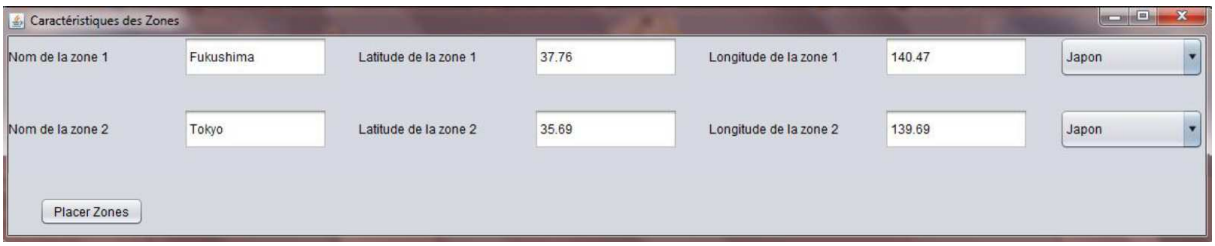

Figure V. 7 Paramétrage des zones, cas à 2 zones

Résultats obtenus :

- Deux curseurs pointant sur les deux villes définies par l'utilisateur.
- Arbre d'une seule branche et de poids non nul :

Poids de l'arbre de Steiner = 0.0471

- Pas de création de zones intermédiaires.
- Les deux zones utilisateurs sont placées sur la carte.

```
Lancement de l'algorithme
**Initialisation de l'algorithme

-> Poids de l'arbre de Steiner = 0.04709093348292085<-
---------------------------------

Fin de l'algorithme

Coordonnees des points de la chaine
----------------------------------------
Zone utilisateur de latitude 35.69 et de longitude 139.69
Zone utilisateur de latitude 37.76 et de longitude 140.47
----------------------------------------
```

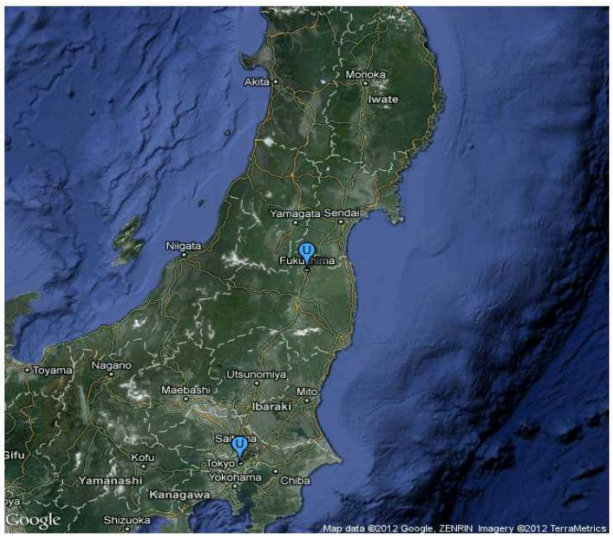

Figure V. 8 Les deux zones utilisateurs

b) Sur deux continents différents

Dans ce cas, on définit deux zones initiales qui sont sur deux continents différents : l'Europe et l'Afrique. L'interface nous demande donc de définir des zones portuaires. On va donc se ramener à deux cas de deux zones chacun.

Données de départ :

Zones initiales	Latitude	Longitude
Paris	48.85	2.35

Zones terminales	Latitude	Longitude
Kairouan	35.67	10.09

Zones portuaires	Latitude	Longitude
Tunis	36.82	10.17
Marseille	43.3	5.37

Tableau V. 3 Données de départ pour le cas 4 zones

Résultats obtenus :

- Quatre curseurs indiquant la zone initiale, terminale et les zones portuaires.
- Deux arbres d'une branche chacun, de poids total non nul.

Poids de l'arbre de Steiner = 0.340

- Pas de création de zones intermédiaires.

```
Lancement de l'algorithme
**Initialisation de l'algorithme

 /!\ Le point de latitude 43.3 et de longitude 5.37 est Steiner

-> Poids de l'arbre de Steiner = 0.340885877141821950.0<-
---------------------------------

Fin de l'algorithme

Coordonnees des points de la chaine
----------------------------------------
Zone utilisateur de latitude 36.82 et de longitude 10.17
Zone utilisateur de latitude 48.85 et de longitude 2.35
Zone utilisateur de latitude 43.3 et de longitude 5.37
Zone utilisateur de latitude 35.67 et de longitude 10.09
----------------------------------------
```

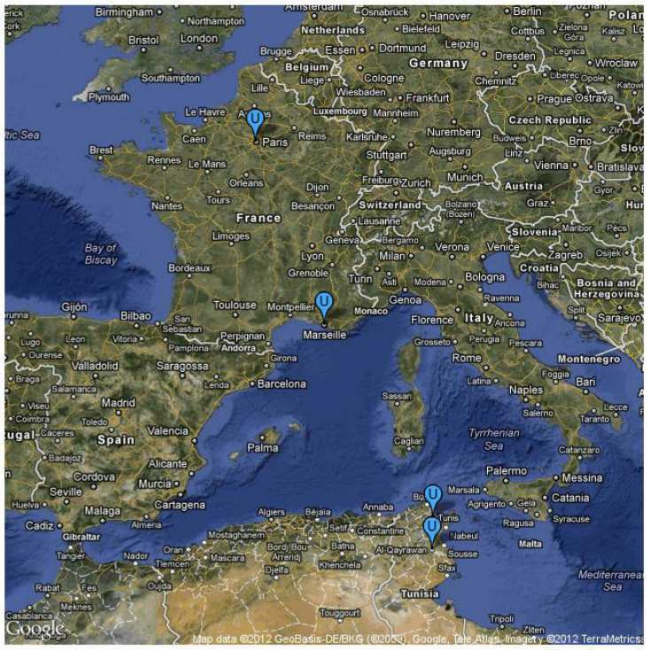

Figure V. 9 Deux zones utilisateurs et deux zones portuaires

V.5.2.2 Deuxième test : cas de 3 zones utilisateurs

a) Sur un même continent

Dans un cas à trois points sur un même continent, l'algorithme doit définir un point de Steiner, donc une zone intermédiaire, permettant de lier nos zones initiales à notre zone terminale, minimisant ainsi la distance totale cumulée de la chaîne logistique de gestion de crises.

Données de départ :

Zones initiales	Latitude	Longitude
Los Angeles	34.05	-118.24

Zones terminales	Latitude	Longitude
Chicago	41.87	-87.6
Nouvelle Orléans	29.95	-90

Zones portuaires	Latitude	Longitude
-	-	-

Tableau V. 4 Données de départ pour le cas 3 zones

Figure V. 10 Paramétrage des zones, cas de 3 zones sur un même continent

Résultats obtenus :

- Création d'une zone intermédiaire équivalente au point de Steiner.
- Quatre curseurs sont placés sur la carte, trois zones U utilisateurs, et une zone I intermédiaire correspondante au point de Steiner.

- Arbre à trois branches de poids non nul :

Poids de l'arbre de Steiner = 0.737

Coordonnées du point de Steiner = (latitude : 34.28 ; longitude : 93.03)

```
Lancement de l'algorithme
**Initialisation de l'algorithme

**Construction d'une zone intermediaire

-> Poids de l'arbre de Steiner = 0.7371481247889499<-
---------------------------------

Fin de l'algorithme

Coordonnees des points de la chaine
----------------------------------------
Zone utilisateur de latitude 29.95 et de longitude -90.0
Zone utilisateur de latitude 41.87 et de longitude -87.6
Zone utilisateur de latitude 34.05 et de longitude -118.24
*Zone generee n1 de latitude 34.28262720869358 et de longitude -
93.0312347768558*
----------------------------------------
```

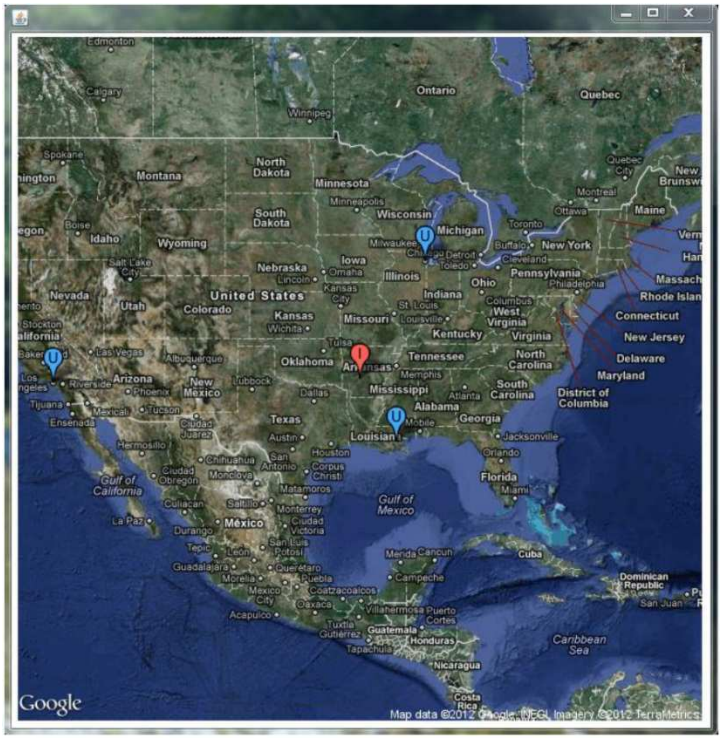

Figure V. 11 Trois zones utilisateurs et une zone intermédiaire

Envisageons désormais une solution « à la main ». Nous choisissons par exemple de placer intuitivement la zone intermédiaire sur Dallas (latitude : 32.80 ; longitude : -96.77), au Sud-Ouest de la zone proposée par l'algorithme. Nous montrons que le poids de l'arbre issu de ce choix (distance totale interzone) est plus élevé que celui de l'arbre issu de l'algorithme.

Zone placée à la main sur Dallas	Zone générée par l'algorithme
Test Steiner ---------- Points composant l'arbre ---------- Point0 de latitude 34.05 et de longitude -118.24 Point1 de latitude 41.87 et de longitude -87.6 Point2 de latitude 29.95 et de longitude -90.0 Point3 de latitude 32.8 et de longitude -96.77 ---------- Liens entre les points ---------- Lien 0 Point de depart de latitude 34.05 et de longitude -118.24 Point d'arrivee de latitude 32.8 et de longitude -96.77 Poids : 0.3756328907603955 Lien 1 Point de depart de latitude 41.87 et de longitude -87.6 Point d'arrivee de latitude 32.8 et de longitude -96.77 Poids : 0.2558006675493297 Lien 2 Point de depart de latitude 29.95 et de longitude -90.0 Point d'arrivee de latitude 32.8 et de longitude -96.77 Poids : 0.13174617238795688 ---------- Poids de l'arbre = 0.7631797306976822	Test Steiner ---------- Points composant l'arbre ---------- Point0 de latitude 34.05 et de longitude -118.24 Point1 de latitude 41.87 et de longitude -87.6 Point2 de latitude 29.95 et de longitude -90.0 Point3 de latitude 34.28 et de longitude -93.03 ---------- Liens entre les points ---------- Lien 0 Point de depart de latitude 34.05 et de longitude -118.24 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.4400242506301015 Lien 1 Point de depart de latitude 41.87 et de longitude -87.6 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.1933708238275402 Lien 2 Point de depart de latitude 29.95 et de longitude -90.0 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.10375305033130817 ---------- Poids de l'arbre = 0.7371481247889499

Nous effectuons le même essai en plaçant la zone intermédiaire à Wichita (latitude : 37.69 ; longitude : -97.34), dans le Kansas, ville qui semble plus centrée que Dallas. Voici ce que l'on obtient :

Zone placée à la main sur Wichita	Zone générée par l'algorithme
Test Steiner ---------- Points composant l'arbre ---------- Point0 de latitude 34.05 et de longitude -118.24 Point1 de latitude 41.87 et de longitude -87.6 Point2 de latitude 29.95 et de longitude -90.0 Point3 de latitude 37.69 et de longitude -97.34 ---------- Liens entre les points ---------- Lien 0 Point de depart de latitude 34.05 et de longitude -118.24 Point d'arrivee de latitude 37.69 et de longitude -97.34 Poids : 0.3731091903189681 Lien 1 Point de depart de latitude 41.87 et de longitude -87.6 Point d'arrivee de latitude 37.69 et de longitude -97.34 Poids : 0.19472975638334014 Lien 2 Point de depart de latitude 29.95 et de longitude -90.0 Point d'arrivee de latitude 37.69 et de longitude -97.34 Poids : 0.20718964254553623 ---------- Poids de l'arbre = 0.7750285892478446	Test Steiner ---------- Points composant l'arbre ---------- Point0 de latitude 34.05 et de longitude -118.24 Point1 de latitude 41.87 et de longitude -87.6 Point2 de latitude 29.95 et de longitude -90.0 Point3 de latitude 34.28 et de longitude -93.03 ---------- Liens entre les points ---------- Lien 0 Point de depart de latitude 34.05 et de longitude -118.24 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.4400242506301015 Lien 1 Point de depart de latitude 41.87 et de longitude -87.6 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.1933708238275402 Lien 2 Point de depart de latitude 29.95 et de longitude -90.0 Point d'arrivee de latitude 34.28 et de longitude -93.03 Poids : 0.10375305033130817 ---------- Poids de l'arbre = 0.7371481247889499

Nous pourrions encore essayer beaucoup de villes des Etats-Unis, sans jamais trouver meilleur positionnement que celui suggéré par l'algorithme. Cet ultime test démontre donc

bien l'intérêt de l'algorithme que nous avons développé du point de vue de la minimisation des distances.

b) Sur deux continents différents

Nous allons tester ici un cas à 3 points comportant une zone initiale sur un premier continent et deux zones terminales sur un deuxième. Nous aurons à créer des zones portuaires. L'application devra alors générer une zone intermédiaire sur le deuxième continent, où trois zones utilisateurs sont finalement implantées, tandis que le cas du premier continent correspondra à un cas à deux points.

Données de départ :

Zones initiales	Latitude	Longitude
Montréal	45.51	-73.55

Zones terminales	Latitude	Longitude
Berlin	52.52	13.41
Sozopol	42.42	27.70

Zones portuaires	Latitude	Longitude
Halifax	44.65	-63.57
La Rochelle	46.16	-1.15

Tableau V. 5 Données de départ pour le cas 2 continents

Résultats obtenus :

- Création d'une zone intermédiaire au niveau du continent européen.
- Six curseurs placés sur la carte, trois zones U utilisateurs, deux zones U portuaires, et une zone I intermédiaire correspondante au point de Steiner.
- Arbre total de poids non nul.

Poids de l'arbre de Steiner = 0.175

Coordonnées du point de Steiner = (latitude : 50.124 ; longitude : 12.931)

```
Lancement de l'algorithme
**Initialisation de l'algorithme

**Construction d'une zone intermediaire

-> Poids de l'arbre de Steiner = 0.175476192968489130.9635527859782873<-
---------------------------------

Fin de l'algorithme

Coordonnees des points de la chaine
----------------------------------------
Zone utilisateur de latitude 42.42 et de longitude 27.7
Zone utilisateur de latitude 52.52 et de longitude 13.41
Zone utilisateur de latitude 46.16 et de longitude -1.15
Zone utilisateur de latitude 45.51 et de longitude -73.55
Zone utilisateur de latitude 44.65 et de longitude -63.57
*Zone generee n1 de latitude 50.124580835225274 et de longitude
12.93175672819148*
----------------------------------------
```

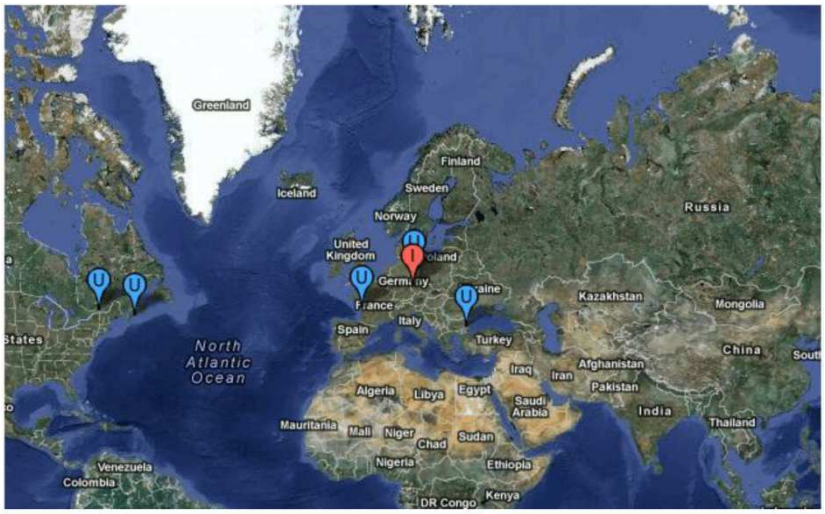

Figure V. 12 Trois zones utilisateurs (2 zones portuaires et une zone intermédiaire)

Bien que cette approche ait prouvé sa pertinence, il est probable qu'en logistique militaire, le choix ne se fasse pas toujours de façon automatisée. En effet, l'intervention humaine est un facteur très important qui se base sur l'expertise et l'expérience des logisticiens aux commandes. Nous proposons alors une aide au positionnement qui permet de prendre en compte des paramètres, autres que la minimisation de la distance, et qui sont caractéristiques de la logistique militaire, comme l'infrastructure en place ou les situations géopolitiques.

V.5.3 Aide au positionnement des zones : Le mode Manuel

Dans cette approche, nous nous mettons sous l'hypothèse qu'il existe des zones d'appuis stratégiques pré installés un peu partout dans le monde. Le logisticien alors, au lieu d'installer la zone logistique dans une région inconnue, va privilégier l'utilisation des zones d'appuis implantées.

L'outil va aider le logisticien dans sa prise de décision pour le choix en nombre et en localisation des zones intermédiaires à utiliser.

Notre choix s'est porté sur le cas du Japon pour illustrer les résultats des simulations. En effet, le séisme du 11 mars 2011, suivi d'un tsunami, a fait de très nombreuses victimes et a provoqué un grave accident nucléaire. Cette catastrophe a nécessité un appui logistique d'urgence considérable. Nous avons donc essayé, en nous basant le plus possible sur ces faits réels, de mettre en place une base de données dans laquelle l'outil d'aide à la décision puise pour ensuite proposer les villes les plus sures et les mieux situées pour pouvoir y implanter des bases logistiques. L'objectif général est d'être le plus efficace possible au niveau des interventions de secours et de soutien à la zone de crise.

V.5.3.1 Données utilisées

Avant de concevoir l'outil d'aide à la décision, il fallait récupérer un maximum de données sur le Japon pour pouvoir proposer un scénario au plus proche de la réalité. Nous avons donc choisi d'utiliser toutes les cartes et les données listées ci-dessous :

Figure V. 13 Carte des villes et aéroports

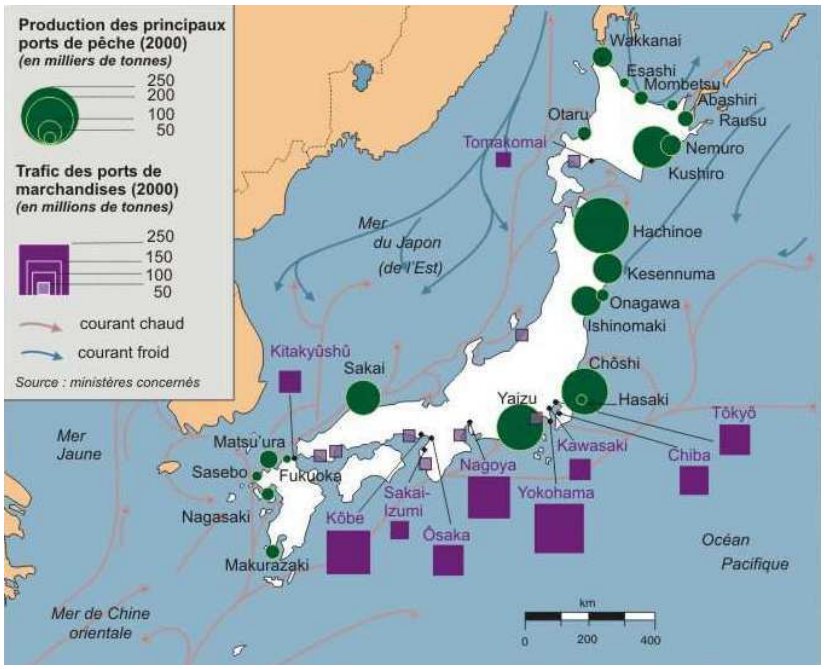

Figure V. 14 Mers et Ports

* *Distance France Japon* : 9 616 Km (de Paris au Japon)

* *Villes choisies* : Fukuoka, Kagoshima, Kobe, Nagoya, Sapporo, Sendai, Tokyo. Ces villes ont été choisis car elles sont suffisamment espacées les unes par rapport aux autres pour couvrir quasiment tout le Japon et car elles comptent parmi les villes les plus grandes du Japon. Elles peuvent donc servir de zones d'appuis stratégiques. Nous avons ensuite effectué une liste des caractéristiques de chaque ville pour pouvoir rentrer ces données dans l'outil d'aide à la décision.

	Fukuoka	Kagoshima	Kobe	Nagoya	Sapporo	Sendai	Tokyo
Nombre de Grandes Routes	4	1	2	4	3	3	5
Nb Aéroports	0	0	0	2	0	0	1
Nb Ports	4	0	1	1	0	1	1
Pondération Montagneuse	1	0	1	3	2	2	2
Pondération Humaine	2	1	2	2	2	1	3
Latitude	33.6	31.6	34.7	35.2	43.1	38.3	35.7
Longitude	130.4	130.6	135.2	136.9	141.4	140.9	139.8

Tableau V. 6 Données relatives aux villes d'appuis stratégiques

Les pondérations montagneuses et humaines varient de 1 à 3 (3 étant l'indice correspondant à une zone montagneuse ou humaine la plus importante).

L'outil propose à l'utilisateur dans un premier temps de saisir le pays qui fait office de zone métropole ainsi que le pays touché par la crise à travers une première interface.

Figure V. 15 Interface d'initialisation

Ensuite lorsque l'utilisateur click sur « OK » l'outil d'aide au positionnement calcule la distance entre ces deux pays et fait une estimation du coût d'affrètement aérien induit si l'on fait un trajet uniquement en avion de la métropole vers la zone de crise (les données sélectionnées sont celles d'un A 380 pour un trajet contenant 550 passagers et des dizaines voire centaines de tonnes de chargement). Dans un souci de simplification, nous avons décidé de comparer les trajets avec un seul moyen de transport, il suffira par la suite de multiplier les coûts par le nombre exact de moyens de transport envoyés. Ce calcul représente le coût que l'utilisateur devrait payer s'il n'implantait pas de bases logistiques intermédiaires. Ce coût est pris en considération dans les calculs de la suite.

V.5.3.2 Résultats des expérimentations

** Premier test*

Nous simulons, dans ce premier test, une situation de crise qui s'est déclaré dans la ville de Sendai. Une fois la ville saisie par l'utilisateur, l'outil affiche les caractéristiques de la ville concernée (nombre de routes, de ports, pondération humaine… cf. partie au-dessus). Un click sur « Annuler » permet un retour à la sélection des pays métropoles et de crise. Enfin lorsque l'utilisateur choisi de « Lancer le calcul » l'outil d'aide au positionnement calcul le coefficient de pertinence pour toutes les villes qui peuvent servir de zones d'appuis stratégiques (sauf celle en crise bien sûr).

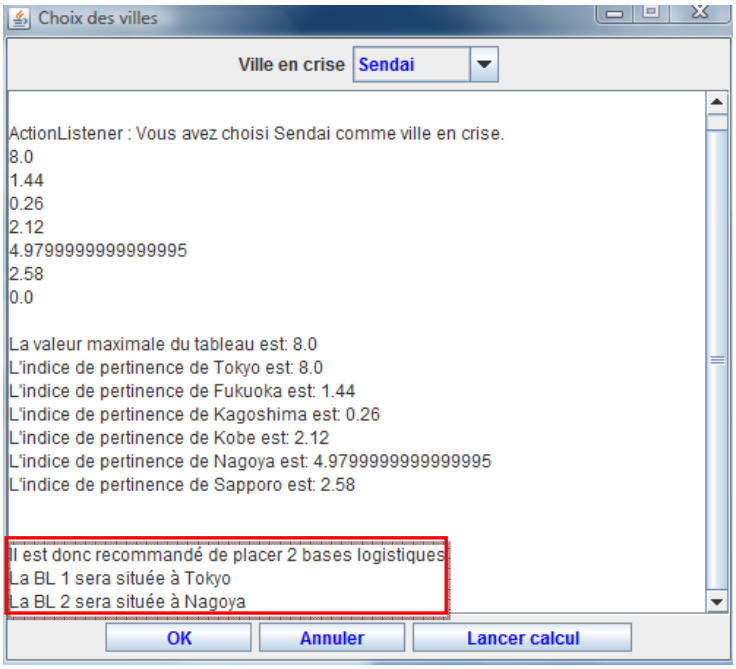

Figure V. 16 Résultat du premier test

Ainsi on voit que si une ville est trop éloignée, ou ne dispose pas d'assez d'infrastructures (ports, routes, humains….) ou si elle reste trop encombrée par des zones montagneuses, le logiciel jugera que si on y implante une base logistique, le coût et l'efficacité de l'intervention ne sera pas optimal. Ainsi l'outil conseille un nombre de bases logistiques à utiliser ou placer (ce nombre reste discutable selon l'urgence et l'état de la zone de crise…) et indique lesquelles choisir en priorité pour optimiser au maximum l'efficacité logistique.

Lors de cette première simulation, l'outil conseille de placer 2 bases logistiques à *Tokyo* et *Nagoya* pour pouvoir aider la ville de *Sendai* très affectée par le séisme.

Nous avons également rajouté un agent perturbateur destructeur de routes et d'infrastructures qui permet de simuler les réalités des crises, à savoir, pour l'exemple du Japon notamment, les lourdes conséquences d'un tsunami ou d'une menace nucléaire (pertes humaines et structurelles…) comme par exemple pour la ville de Tokyo où les conséquences sont une destruction de 3 routes suite au tsunami, de nombreuses pertes humaines et une réquisition de l'aéroport pour permettre aux personnes restantes de fuir le pays :

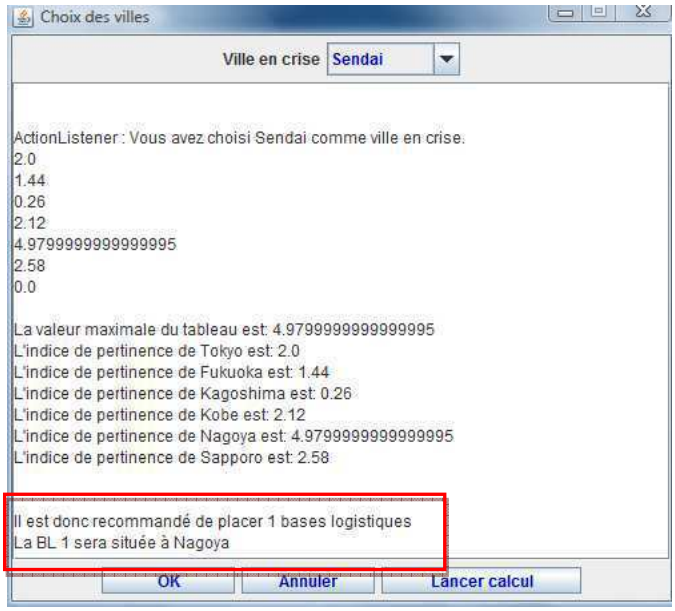

Figure V. 17 Cas d'une perturbation

L'outil d'aide à la décision s'adapte. Dans le cas d'une crise survenant dans une crise (exemple : lorsqu'une ville a été complètement détruite par une catastrophe ou une attaque) il ne conseille pas au logisticien d'y implanter une base. Dans notre exemple en redémarrant une simulation, le logiciel calcule le nouvel indice de pertinence de *Tokyo* qui a baissé jusqu'à 2 (au lieu de 8 précédemment) et ne conseille plus qu'une seule BL placée à *Nagoya*.

** Second test*

Dans ce deuxième jeu de simulation, c'est la ville de *Tokyo* qui a été frappée par le sinistre.

Nous obtenons les résultats suivants : L'outil d'aide au positionnement propose de placer 3 zones intermédiaires, à savoir *Nagoya* en premier lieu, *Sendai* en deuxième lieu et finalement *Kobe* en troisième position

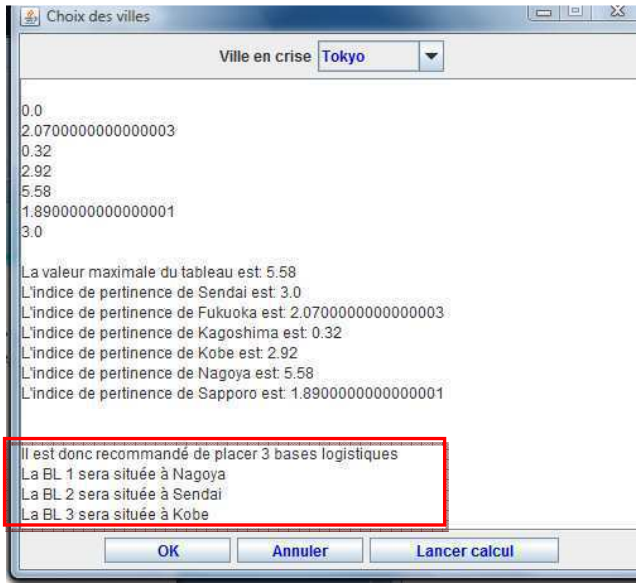

Figure V. 18 Résultat du second test

Un deuxième mode de fonctionnement possible et de laisser l'utilisateur choisir la zone d'appui stratégique et lui fournir une évaluation sur son choix.

Ainsi, avec *Sendai* comme ville de crise, si le commandement souhaite choisir *Tokyo* comme Base Logistique, le logiciel lui confirme que c'est un très bon choix (le meilleur même). S'il choisit *Kobe* c'est un choix correct. Par contre s'il choisit (pour des raisons politiques par exemple) *Kagoshima* qui est à l'opposé de l'île et qui ne dispose pas d'infrastructures très intéressantes, le logiciel l'avertit que de lourdes conséquences financières sont à prévoir.

Figure V. 19 Evaluation de la décision du logisticien

Nous constatons que notre approche permet à la fois de jouer le rôle d'outil d'aide à la décision pour le commandement militaire mais aussi d'anticiper le positionnement stratégique et géographique de bases logistiques dans une zone en crise. Ce dernier peut très facilement être adapté à la logistique civile, où il suffirait d'adapter les critères de calcul de pertinence.

Dans la section suivante, nous présentons les résultats de simulations de l'agent holonique Need Estimating Agent, qui permet d'estimer les consommations futures de marchandises au niveau des zones sinistrées.

V.5.4 Les tests du Need Estimating Agent

Pour réaliser ces simulations, deux séries de tests ont été effectuées.

V.5.4.1 Premier test : variation linéaire en fonction du nombre de personnes

Lors du premier test, nous avons considéré les paramètres suivants :

- Une variation linéaire du nombre de personnes.

- Conditions météo stables : 20°, 90% d'humidité

- Optimisation faite à partir de la température, du degré d'humidité et du nombre de personnes présentes sur le site.

- On suppose les variations linéaires en fonction du nombre de personnes.

- On suppose que pour le matériel étudié, le débit moyen est de 100 unités par semaine pour 100 personnes. Ce débit varie en fonction de la température sur le site et de l'humidité (données représentatives des conditions météo). Cette variation est calculée à partir d'une loi basée sur la logique floue.

- Le site a une durée de vie de huit semaines. Les deux premières semaines représentent l'établissement du site, et les deux dernières, son évacuation.

Nous obtenons les résultats du tableau V.7. Les résultats présentés ici sont comparés avec les résultats obtenus avec la première version du NEA, notamment en termes d'erreur obtenue. Avec le nouvel estimateur à base d'ARMAX, l'erreur tend à s'annuler plus rapidement qu'avec les estimateurs statistiques linéaires anciennement développés.

semaine	Nombre de personnes	Estimation logique floue	Valeur réelle	Estimation ARMAX	Statistical estim. error	ARMAX error
1	500	486,26	450	475,67	7,45%	5,39%
2	1500	1458,79	1350	1397,37	7,45%	3,38%
3	2000	1945,06	1700	1757,54	4,92%	3,27%
4	3000	2917,59	2750	2795,171	3,59%	1,61%
5	2500	2431,32	2250	2276,511	2,96%	1,16%
6	2500	2431,32	2300	2322,701	1,57%	0,96%
7	1300	1264,29	1150	1154,842	0,97%	0,39%
8	700	745,4	800	798,67	0,38%	0,25%

Tableau V. 7 Tableau de valeurs du premier test

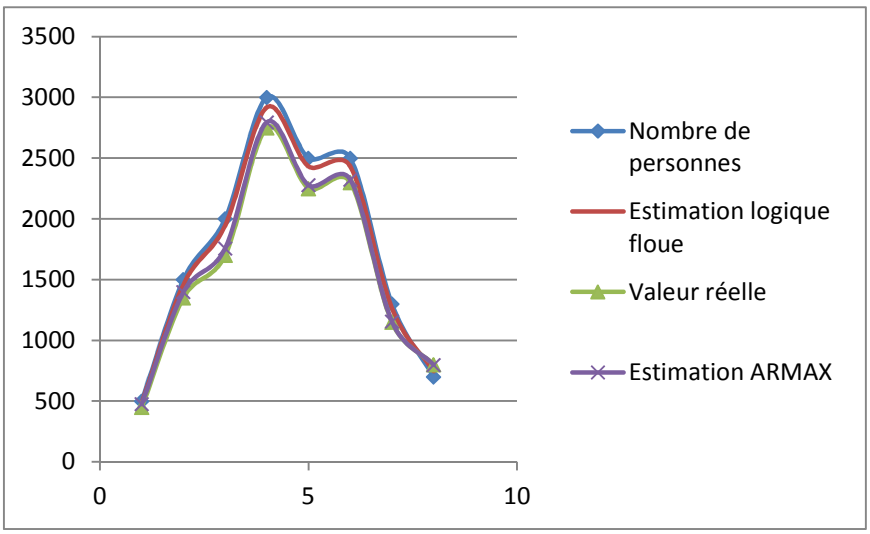

Figure V. 20 Courbe de valeurs du premier test

V.5.4.2 Second test : variation non linéaire en fonction du nombre de personnes

Une seconde série de test a été réalisée, les conditions de ce second test sont identiques à une exception près : Les variations de consommation ne sont plus linéaires en fonction du nombre de personnes présentes sur le site.

La durée du site est de 11 semaines.

semaine	Nombre de personnes	Estimation logique floue	Valeur réelle	Estimation ARMAX	Statistical estim. error	ARMAX error
1	5000	4862,64	4500	4752,39	7,45%	5,30%
2	15000	14587,93	15500	15190,04	8,05%	2,04%
3	20000	19450,57	18000	18475,4	3,92%	2,05%
4	30000	29175	20000	22327,23	19,15%	11,66%
5	25000	24313	19000	20220,511	9,96%	6,03%
6	25000	24313	19000	20220,701	9,57%	6,03%
7	13000	12643	14000	13594,69	7,97%	2,98%
8	7500	7294	7500	7458,16	4,38%	0,56%
9	5000	4862,64	4500	4524,48	2,15%	0,52%
10	15000	14187,93	15500	15279,3	2,36%	1,44%
11	20000	19450,57	18000	17896	1,09%	0,58%

Tableau V. 8 Tableau de valeurs du second test

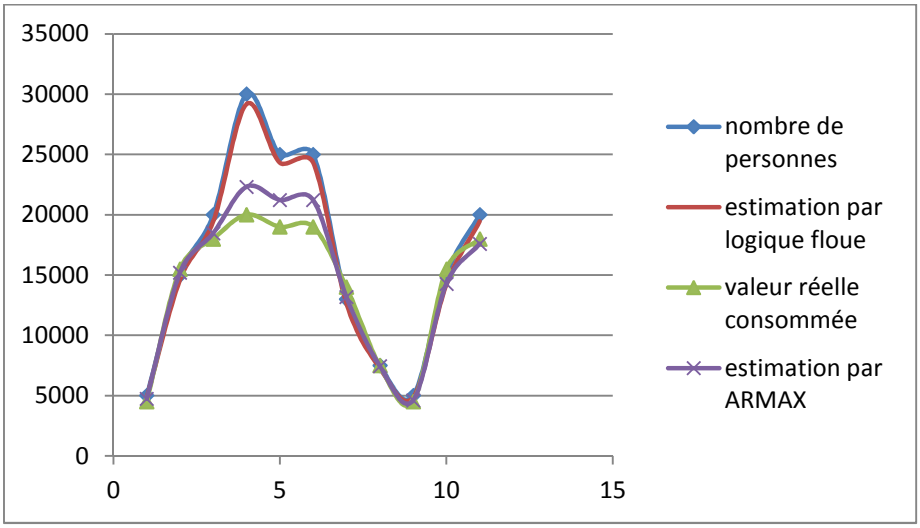

Figure V. 21 Courbe de valeurs du second test

Dans le cas du deuxième test, le graphe (Figure V.21), et le tableau de valeurs, montrent bien l'évolution de l'estimation au cours du temps. On voit très nettement l'estimateur ARMAX s'améliorer à chaque fois, tout en prenant le pas sur l'estimation linéaire (régression linéaire). L'estimation s'affine au cours du temps jusqu'à coller pratiquement à la réalité et donner une estimation précise de la valeur recherchée (moins de 2% d'erreur).

Nous présentons dans la section suivante les résultats des simulations de l'approche proposée pour l'ordonnancement distribué des tâches de livraisons.

V.5.5 Comportement des agents ordonnanceurs : Scénario Japon

Notre application est le résultat d'un travail important et continu entre notre équipe de recherche OSL (Optimisation des Systèmes Logistiques- LAGIS - ECLille) et le département logistique de EADS, dans le but de mettre en œuvre un démonstrateur logistique pour la gestion des flux d'une chaîne de gestion de crise.

La clause de confidentialité qui lie le LAGIS / EC-Lille et EADS, empêche la diffusion des résultats de données réelles. Ainsi, afin d'évaluer la solution proposée dans le présent mémoire et valider l'approche de résolution du problème d'ordonnancement distribué, nous avons appliqué la méthodologie proposée sur un exemple académique d'une chaîne logistique de gestion de crise. Pour expliquer concrètement le fonctionnement de la solution proposée, nous allons considérer l'exemple de la crise engendrée par le séisme qui a touché le Japon en 2011.

V.5.5.1 Données du scénario

Ce scénario dure 3 semaines :

- Semaine 1: Situation d'urgence due aux inondations ;
- Semaine 2: Retour à un état stationnaire ;
- Semaine 3: Nouvelle situation d'urgence à cause des risques nucléaires.

Nous considérons au total 9 zones sinistrées, une zone métropole et 3 zones intermédiaires, pour avoir un problème d'ordonnancement un peu plus riche.

Nous avons considéré que l'apparition de la crise détectée par l'agent métropole conduit à la création de 3 zones intermédiaires : ZI1, ZI2 et ZI3 (Figure 14). Chaque agent zone intermédiaire reçoit 3 tâches à ordonnancer pour aider les zones sinistrées.

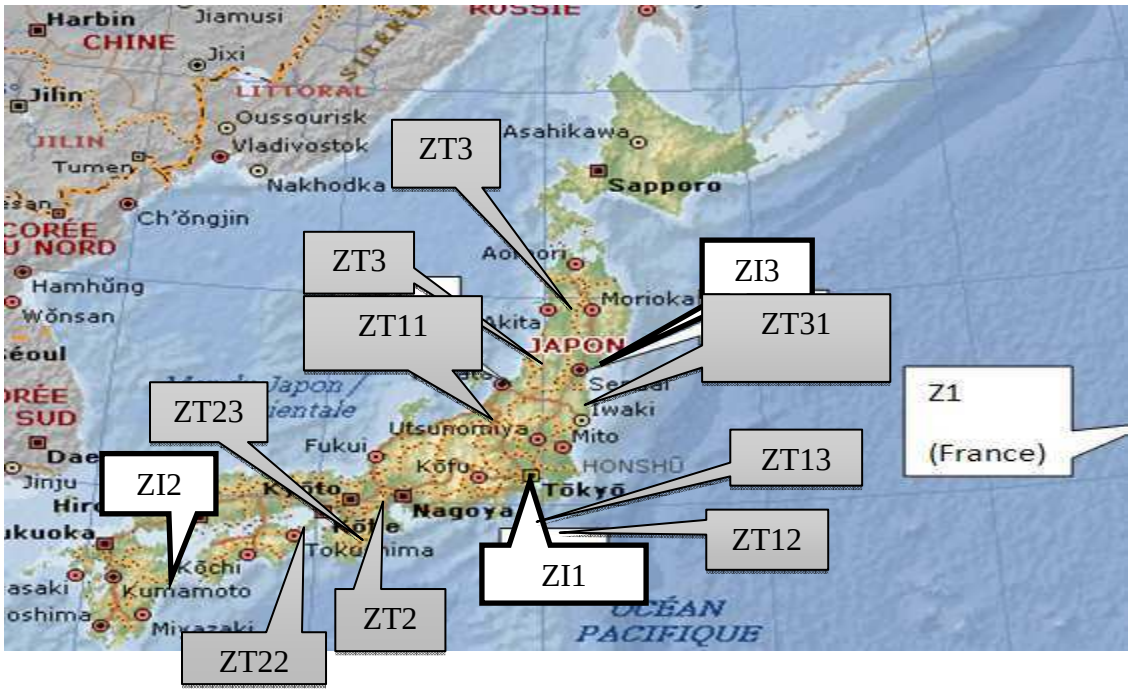

Figure V. 22 Mise en place de la chaîne logistique

Les moyens de transport utilisés sont les suivants :

*Avions transporteurs (caractéristiques basées sur l'Airbus A400M, en développement chez EADS)

* Camions logistiques (caractéristiques basées sur le WPK 440, 4x4 logistique de l'armée de terre française)

Le tableau ci-dessous représente les caractéristiques de ces moyens de transport.

	Airbus A400M	WPK 440
Capacité de chargement	340 m3	50 m3
Vitesse	850 Km /h	70 Km/h
Carburant	Jet A1 : 1 €/L	Gasoil : 1,4 € /L
Consommation horaire	17 000 L	10 L
Cout horaire	17 000 €	14 €

Tableau V. 9 Caractéristiques des moyens de transport utilisés pour la CLGC

Pour les délais de livraison, nous supposons que les délais de transport entre zone intermédiaire et zone terminale est de 3 jours par voie terrestre et 1 jour par voie aérienne. Les délais de transport entre zone métropole et zone intermédiaire est de 2 jours par voie terrestre et 1 jour par voie aérienne. Dans notre cas le transport entre la zone métropole (France) et les zones intermédiaires (situées au Japon) ne peut s'effectuer que par voie aérienne.

Le temps d'exécution correspond au champ « temps d'exécution » de la tâche de livraison. Ce champ va être mis à jour à chaque fois que nous changeons le moyen de transport pour le ré ordonnancement.

Comme il est difficile de donner une valeur exacte pour les coûts de référence, les valeurs utilisées lors de ces simulations sont des valeurs approximatives (cela dépend du lieu, de l'intensité de la crise, de la nature de la marchandise etc.)

V.5.5.2 Première semaine de déploiement

Afin de mettre en évidence la variation des coûts de livraison, nous considérons que les agents zones ordonnanceurs reçoivent des requêtes trois fois au cours de cette semaine (au jour 1, jour 4 et jour 7).

Nous considérons tout d'abord les données suivantes pour les marchandises gérées :

- R1 : vêtements
- R2 : nourriture
- R3 : médicaments

Initialement nous avons les quantités initiales de ressources illustrées à la figure V.23 suivante.

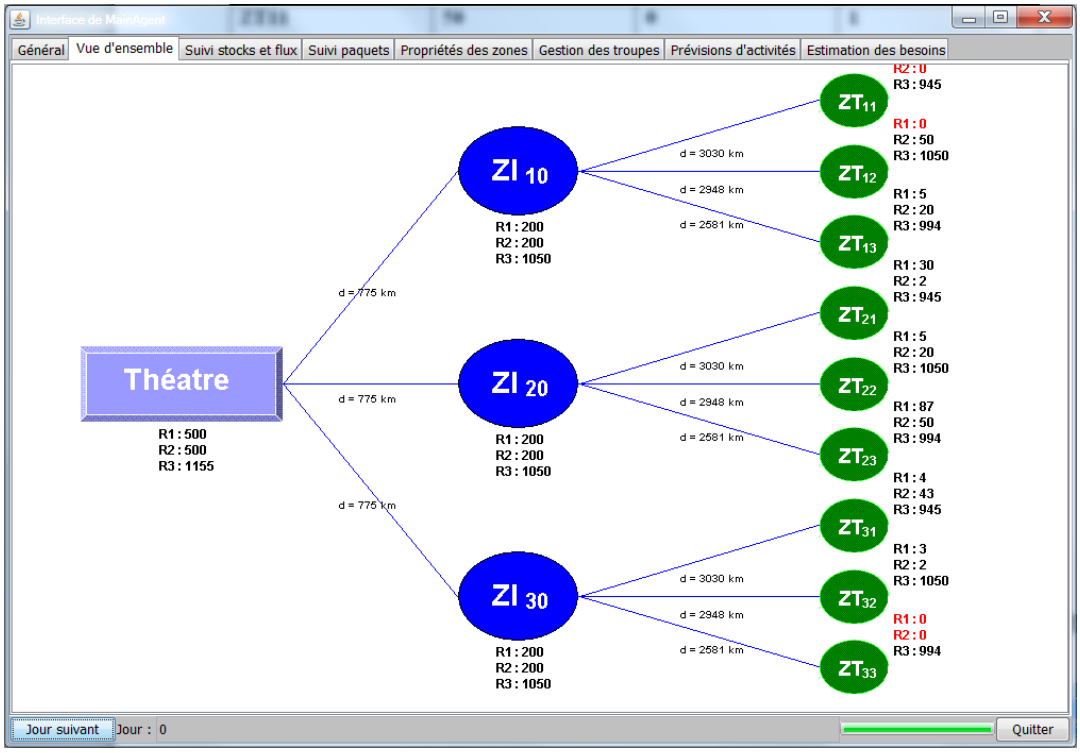

Figure V. 23 Vue d'ensemble de la chaîne logistique

A t= jour 1, les trois agents ZI1, ZI2 et ZI3 reçoivent l'ensemble des requêtes suivantes :

- *V1 :{ émetteur : ZT11, receveur : ZI1, besoin : R2, quantité : 5)*
- *V2 :{ émetteur : ZT12, receveur : ZI1, besoin : R3, quantité : 20)*
- *V3 :{ émetteur : ZT12, receveur : ZI1, besoin : R1, quantité : 5)*
- *V4 :{ émetteur : ZT21, receveur : ZI2, besoin : R3, quantité : 5)*

- *V5 :{ émetteur : ZT22, receveur : ZI2, besoin : R3, quantité : 20)*
- *V6 :{ émetteur : ZT23, receveur : ZI2, besoin : R3, quantité : 5)*
- *V7 :{ émetteur : ZT33, receveur : ZI3, besoin : R1, quantité : 5)*
- *V8 :{ émetteur : ZT33, receveur : ZI3, besoin : R2, quantité : 20)*
- *V9 :{ émetteur : ZT33, receveur : ZI3, besoin : R3, quantité : 5)*

Les quantités de ressources demandées sont basées sur les estimations fournis par l'agent Need Estimating Agent (Cf. chapitre III).

La Figure V.24 illustre l'ensemble des tâches reçues par les agents zones intermédiaires au premier jour de la crise.

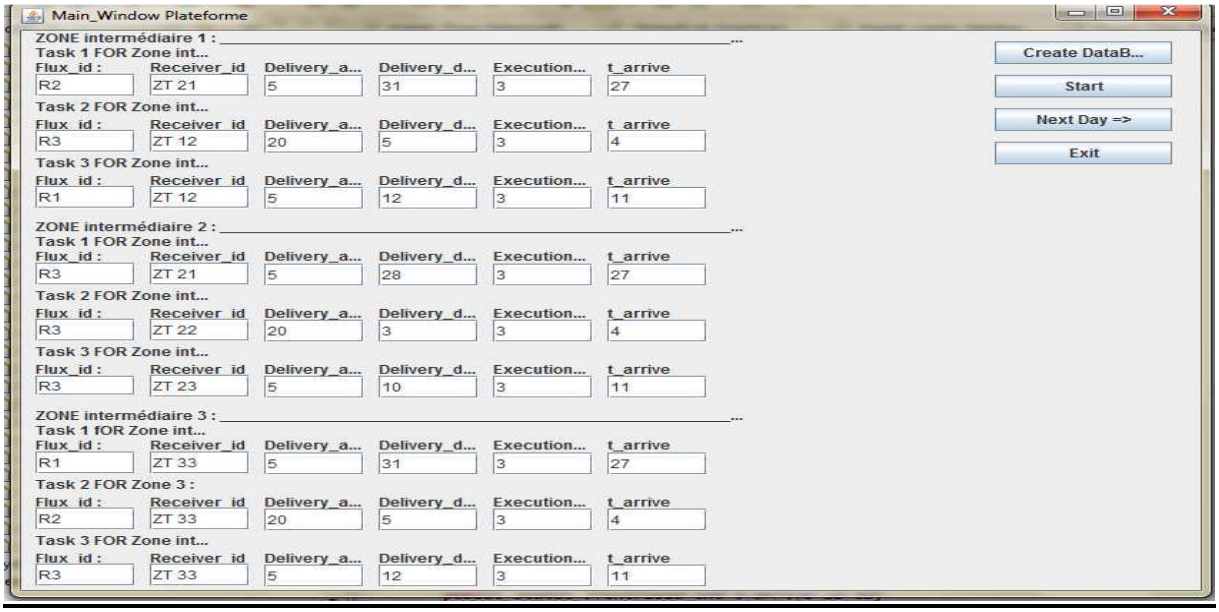

Figure V. 24 Tâches reçues par les agents ordonnanceurs au 1er jour (semaine 1)

Les données reportées dans la figure V.24 seront transmises aux agents zones ordonnanceurs qui ont pour mission de générer les plans de livraison.

Chaque agent ordonnanceur concerné récupère les informations relatives à la requête reçue et procède à l'ordonnancement selon la date d'arrivée des tâches et en respectant la date de livraison demandée. L'agent Intégrateur Evaluateur évalue alors les coûts engendrés par les plannings générés. Les indicateurs de performance obtenus sont comparés aux valeurs de référence. Leurs valeurs sont les suivantes ;

Local Schedule	**Indicator1**		**Indicator2**		**Indicator3**		**Indicator4**	
	C^{routes}	C^{routes}_{Ref}	$C^{transport}$	$C^{transport}_{Ref}$	$C^{earliness}$	$C^{earliness}_{Ref}$	C^{delay}	C^{delay}_{Ref}
ZI1	1,1	2,5	11,3	30	2,7	2	55	15
ZI2	1,97	2,5	16,4	30	4,2	2	30	15
ZI3	2,7	2,5	9.4	30	1,6	2	25	15

Tableau V. 10 Coûts obtenus en K€

En pratique, nous permettons un pourcentage de dépassement des coûts des ordonnancements par rapports aux coûts fixés. Donc un plan ne sera pas directement rejeté si son coût est supérieur au coût de référence mais si le pourcentage de dépassement est supérieur à un seuil fixé. L'agent Intégrateur Evaluateur détecte le dépassement au niveau de l'agent ZI1. Un ordre de ré-ordonnancement est généré. L'agent ordonnanceur ZI1 est notifié de la demande de ré ordonnancement, et envoie une demande à l'agent Transport pour changer le moyen de transport.

Les indicateurs de performance obtenus après le ré ordonnancement sont les suivants :

Local Schedule	**Indicator1**		**Indicator2**		**Indicator3**		**Indicator4**	
	C^{routes}	C^{routes}_{Ref}	$C^{transport}$	$C^{transport}_{Ref}$	$C^{earliness}$	$C^{earliness}_{Ref}$	C^{delay}	C^{delay}_{Ref}
ZI1	0	2,5	16,7	30	2,7	2	0	15
ZI2	1,97	2,5	8,4	30	4,2	2	30	15
ZI3	2,7	2,5	9.4	30	1,6	2	25	15

Tableau V. 11 Coûts (K€) obtenus après ajustement

Après avoir quitté l'état critique d'urgence, des requêtes apparaissent de nouveau au troisième jour de la crise. La figure ci-dessous montre les différentes tâches reçues par les agents ordonnanceurs.

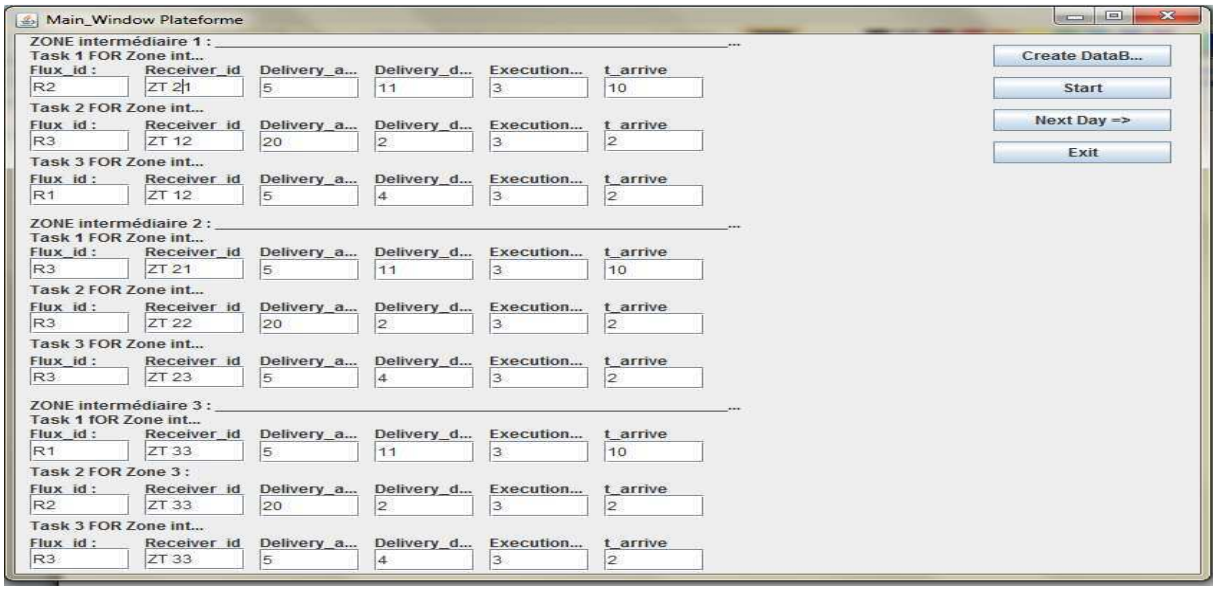

Figure V. 25 Liste des tâches reçues par les agents ordonnanceurs au 3ème jour (semaine 1)

Nous remarquons que dans ce cas, la deuxième tâche et la troisième tâche sont arrivées en même temps, mais la priorité est donnée à la deuxième tâche car elle possède la date de livraison la plus imminente. Après simulation nous obtenons :

- Coût de retard de livraison = 43200 € > Coût de retard de référence
- Cout d'avance de livraison=0 € < Coût d'avance de référence

Ayant effectué le changement du moyen de transport pour le ré ordonnancement, les nouvelles valeurs des indicateurs de performance sont *4800 €* pour le retard et *0 €* pour l'avance.

A t = jour 5, les trois agents ZI1, ZI2 et ZI3 reçoivent l'ensemble des requêtes suivantes (Figure V.24).

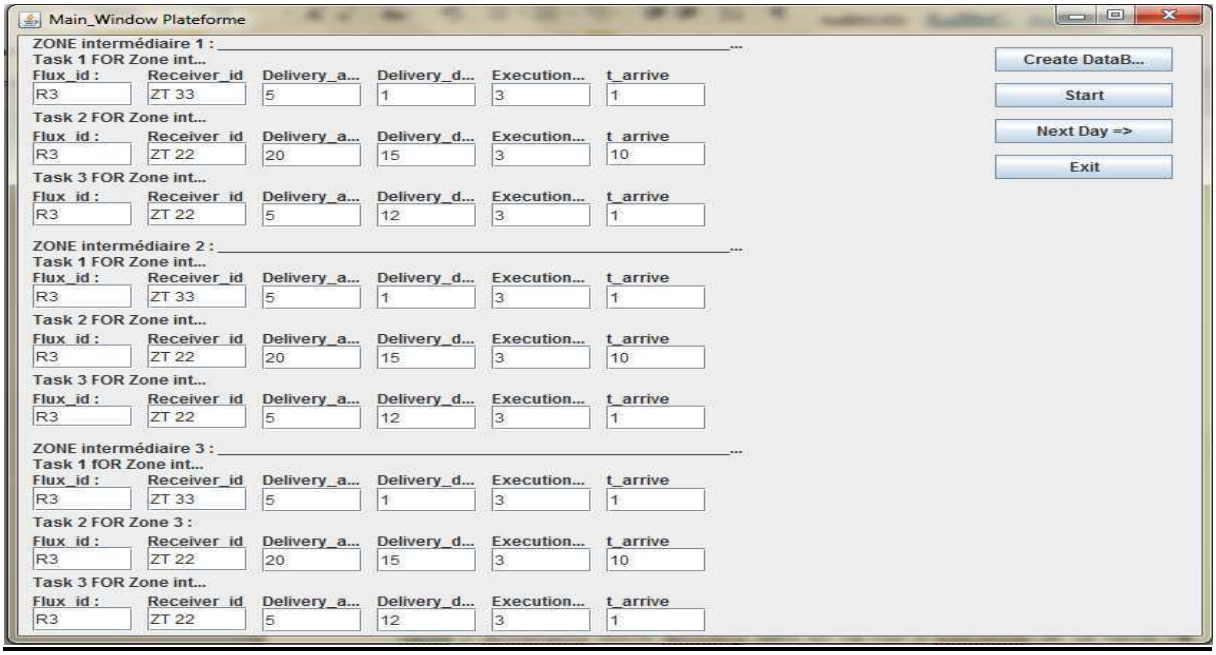

Figure V. 26 Liste des tâches reçues par les agents ordonnanceurs au 5ème jour (semaine1)

Les plannings de livraison génèrent dans ce cas (après le ré ordonnancement) à ;

- Coût de retard de livraison = 4800 € <Coût de retard de référence
- Cout d'avance de livraison = 2020 €< Coût d'avance de référence

V.5.5.3 Deuxième semaine de déploiement

Au cours de cette semaine, il y a un retour à un état stationnaire et nous quittons l'état d'urgence.

V.5.5.4 Troisième semaine de déploiement

Nous revenons à état d'urgence. En effet, le séisme puis les inondations ont entrainé un risque nucléaire. Nous devons procéder à une évacuation générale des ressources humaines présentes au Japon : c'est ce qu'on appelle la rétro-logistique.

*Nous considérons les données suivantes :

- R1 : soldats,
- R2 : médecins,
- R3 : médicaments

	R1	R2	R3
Zone métropole	500	500	500
ZI1	195	180	150
ZI2	200	200	170
ZI3	195	180	150
ZT11	50	5	1
ZT12	5	50	20
ZT13	5	20	3
ZT21	30	2	5
ZT22	5	20	20
ZT23	87	50	5
ZT31	4	43	7
ZT32	3	2	10
ZT33	5	20	5

Tableau V. 12 Les quantités initiales de ressources au début de la troisième semaine

Au premier jour de cette semaine, les requêtes à traiter sont comme suit;

- *V1 :{ émetteur : ZT11, receveur : ZI1, besoin : R2, quantité : 5)*
- *V2 :{ émetteur : ZT12, receveur : ZI1, besoin : R3, quantité : 20)*
- *V3 :{ émetteur : ZT12, receveur : ZI1, besoin : R1, quantité : 5)*
- *V4 :{ émetteur : ZT21, receveur : ZI2, besoin : R3, quantité : 5)*
- *V5 :{ émetteur : ZT22, receveur : ZI2, besoin : R3, quantité : 20)*
- *V6 :{ émetteur : ZT23, receveur : ZI2, besoin : R3, quantité : 5)*
- *V7 :{ émetteur : ZT33, receveur : ZI3, besoin : R1, quantité : 5)*
- *V8 :{ émetteur : ZT33, receveur : ZI3, besoin : R2, quantité : 20)*
- *V9 :{ émetteur : ZT33, receveur : ZI3, besoin : R3, quantité : 5)*

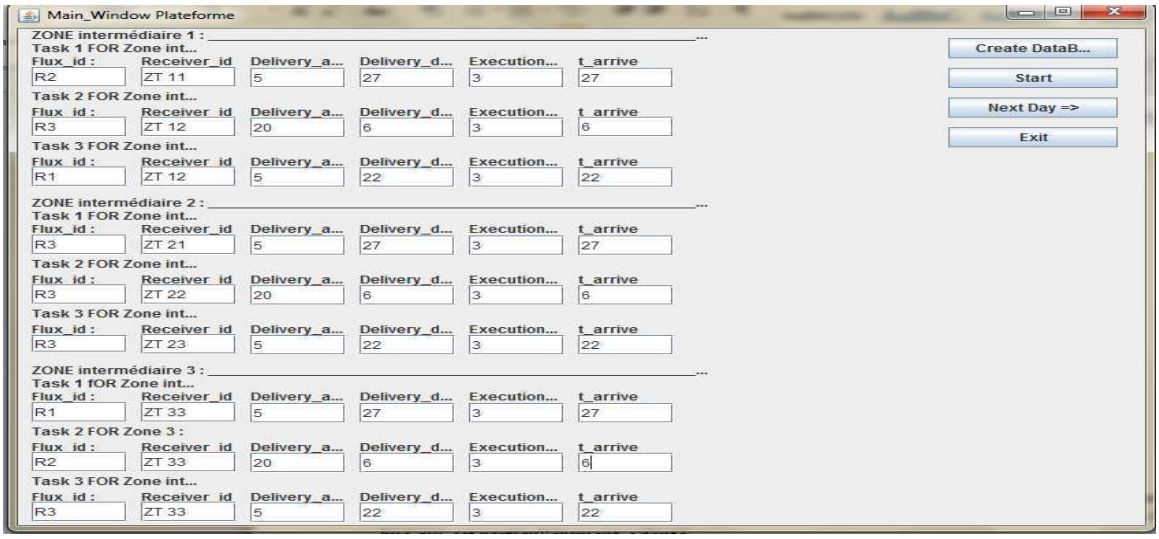

Figure V. 27 Liste des tâches reçues par les agents ordonnanceurs au 1er jour (semaine 3)

L'algorithme d'ordonnancement génère les coûts suivants ;

- Coût de retard de livraison = 12250 € >Coût de retard de référence
- Cout d'avance de livraison=0 € < Coût d'avance de référence

A cause de la gravité de la crise et afin de ne pas entamer les stocks de sécurité, les zones sinistrées ont besoin de plus de ressources pour ne pas envisager une nouvelle situation critique. La figure V.28 définit l'ensemble des tâches créées.

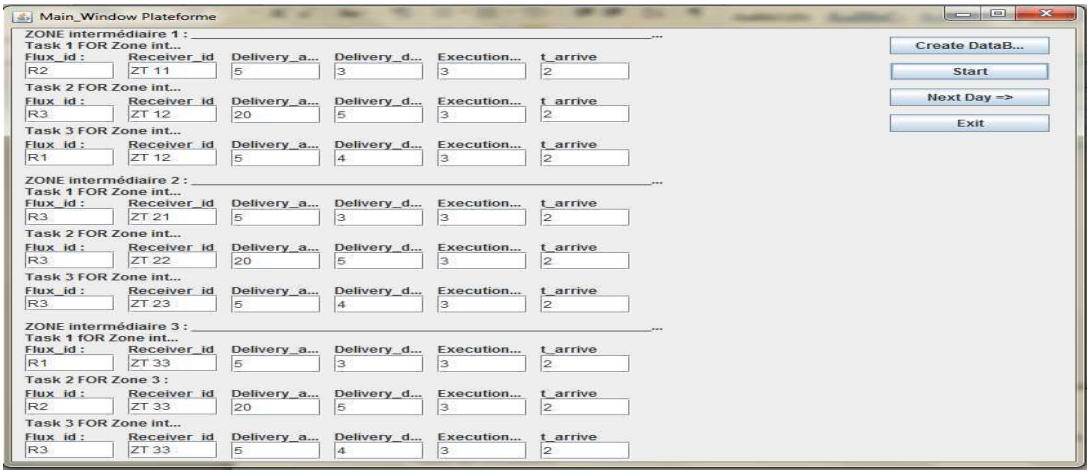

Figure V. 28 Liste des tâches reçues par les agents ordonnanceurs au 2ème jour (semaine 3)

Au 5ème jour de cette semaine (Figure V.29) les tâches reçues sont les suivantes :

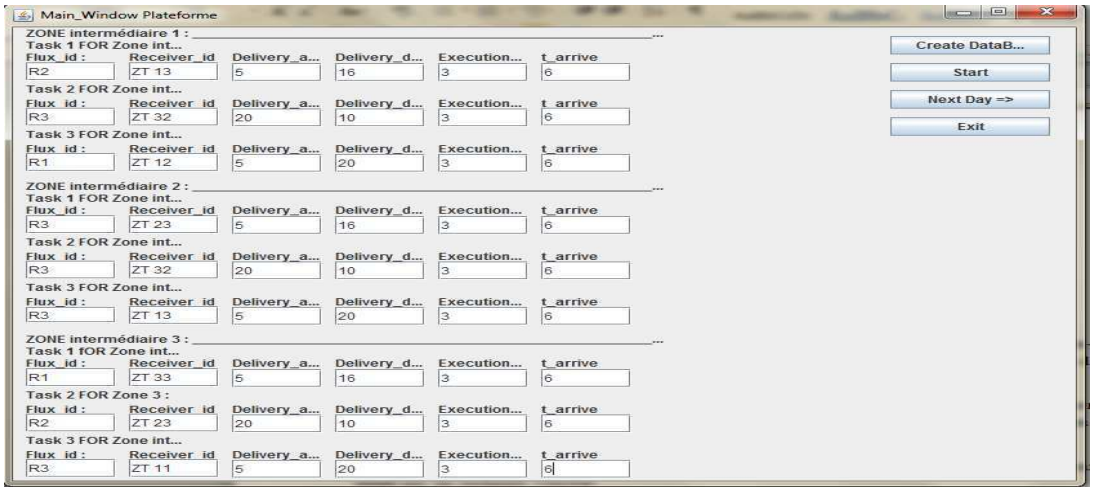

Figure V. 29 Liste des tâches reçues par les agents ordonnanceurs au 5ème jour (semaine 3)

➔Un ordonnancement sans entente (sans coopération entre agents) donne :

- Coût de retard de livraison = 0 € <Coût de retard de référence
- Cout d'avance de livraison = 5260 € < Coût d'avance de référence

Nous avons comparé les coûts obtenus avec des coûts que peut donner un algorithme d'ordonnancement sans entente et non adaptatif en cours de simulation.

	Coût total obtenu avec entente (K€)	Coût total selon un algorithme sans entente (K€)
Jour1_Semaine1	19400	25 440
Jour3_Semaine1	9800	28 200
Jour5_Semaine1	7 920	22 200
Jour1_Semaine3	13 400	27 200
Jour2_Semaine3	9200	31 600
Jour5_Semaine3	7 400	24 600

Tableau V. 13 Comparaison entre le coût de livraison total obtenu, avec et sans entente entre les agents

Les valeurs du tableau précédent permettent de tracer les courbes ci-dessous.

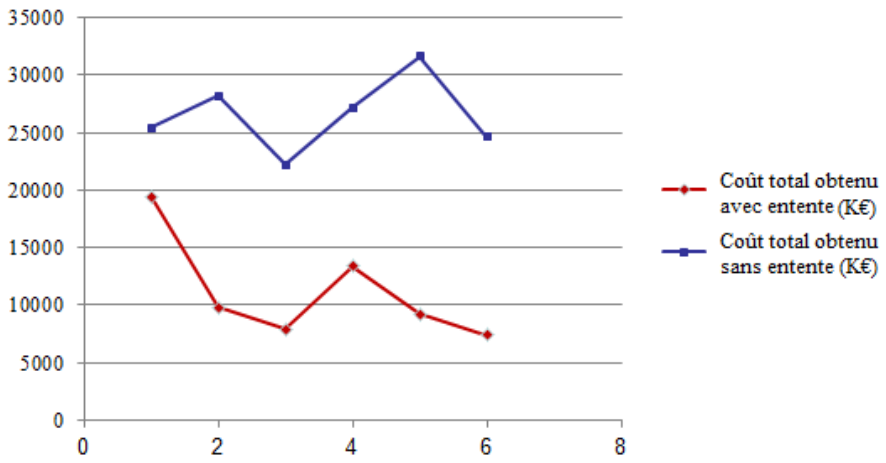

Figure V. 30 Courbes des coûts de livraison

La Figure V.30 montre la grande différence entre le coût de livraison obtenu avec l'approche de résolution distribuée proposée et celui obtenu en appliquant un algorithme d'ordonnancement en absence d'entente entre les agents. Dans ce dernier cas, les agents travaillant en boucle ouverte et sans un vrai retour sur la qualité de leurs solutions, engendrent un coût de livraison beaucoup plus important. D'où l'efficacité de la solution proposée ainsi que celle de l'approche multi-agents dans la résolution d'un problème d'ordonnancement dans une chaîne logistique assez complexe.

L'agent Sniffer de la plateforme Jade (figure V.31) permet de tracer l'échange de messages entre les différents agents du système.

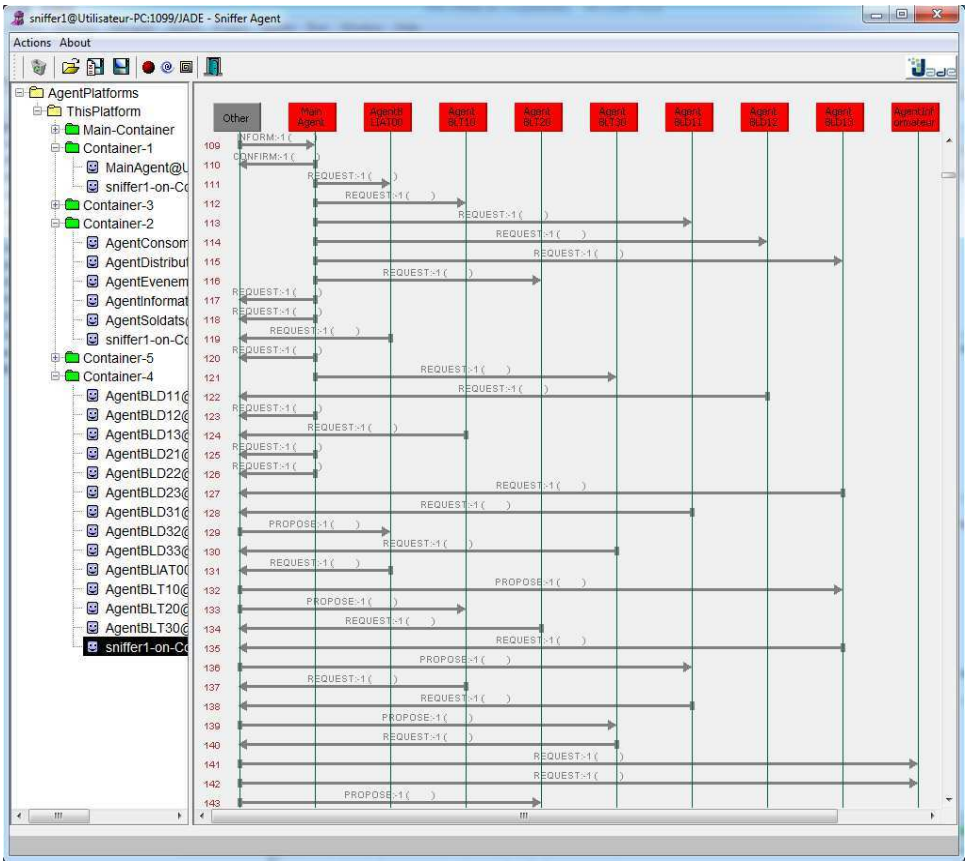

Figure V. 31 Sniffer Agent

Au bout de 14 jours de simulation, nous comparons l'évolution des stocks en activant le mode d'optimisation de l'ordonnancement, avec les résultats obtenus avec le démonstrateur dans sa version antérieure. Nous remarquons que pour le même jeu de simulation, notre approche permet de lisser l'évolution des stocks, en évitant de tomber sous le seuil de sécurité, comme c'était le cas avec la première version.

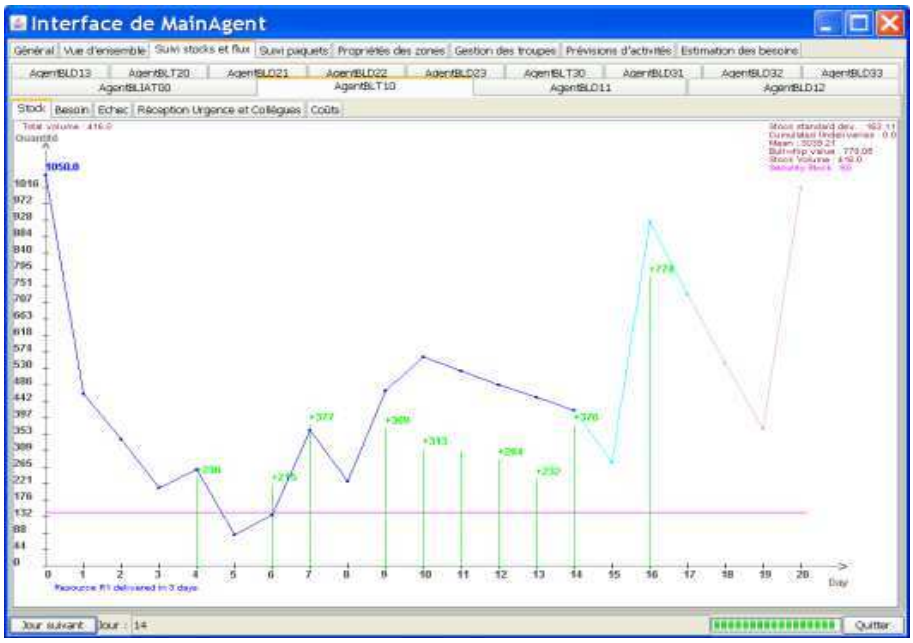

Figure V. 32 Courbe des stocks avec première version

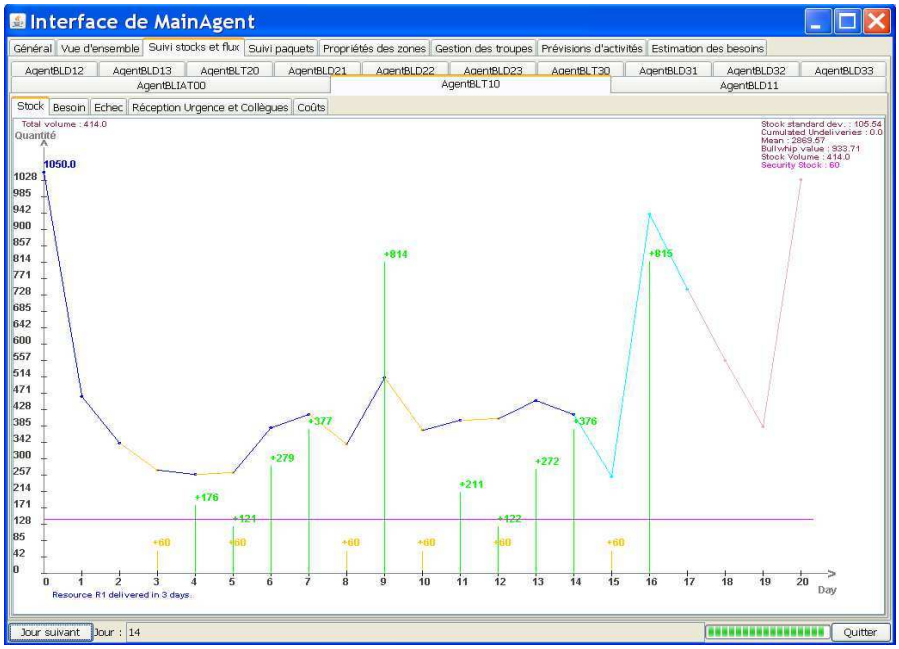

Figure V. 33 Courbe des stocks avec l'ordonnancement distribué

V.5 Conclusion

Dans ce chapitre, nous avons simulé le fonctionnement des approches d'optimisation adoptées dans notre démonstrateur OBAC. Nous avons commencé par présenter le fonctionnement général du démonstrateur. Ensuite nous avons présenté plusieurs simulations pour évaluer la performance de notre système pour la gestion de crise. Les résultats et scénarii de simulation montrent la flexibilité et l'adaptabilité du système face aux aléas de la crise. L'OBAC permet certainement d'aider les décideurs logisticiens dans leurs prises de décision mais, face à leurs

expertises il ne sera pas forcément suivi. C’est pour cette raison qu’il doit s’adapter en temps réel et se réajuster à la réalité le plus finement possible afin de proposer des solutions cohérentes.

Conclusion Générale

Dans beaucoup de secteurs industriels et notamment celui de la gestion de crise, les chaînes logistiques sont soumises à des aléas importants (demandes incertaines, désastres naturels ou causés par l'homme, etc.) qui menacent leurs évolutions et dégradent souvent leurs performances.

Dans ce contexte, le domaine de gestion des chaînes logistiques a suscité, ces dernières années, un grand intérêt. Les travaux de recherche présentés dans cette thèse, dans le cadre d'une coopération avec le département logistique d'EADS, visent à apporter une réponse à ces défis, dans une démarche qui allie les méthodes d'optimisation au paradigme multi-agent.

Dans ce mémoire, nous nous intéressons aux solutions d'aide à la décision afin de faciliter le travail des décideurs logisticiens en cas de crise. L'approche orientée agents nous étant apparue pertinente, notamment pour la prise en compte de comportements individuels au niveau des différents acteurs de la chaîne, nous avons apporté, dans le cadre de cette thèse, des solutions basées sur une approche de modélisation, optimisation et simulation orientée agents de flux logistiques, dans un contexte à fortes perturbations, plus précisément, dans le cas d'une chaîne logistique de gestion de crise.

Notre contribution a couvert trois volets d'investigation : un module d'aide au positionnement des zones logistiques qui permet de minimiser les distances entre les zones et d'optimiser leurs nombre et ce pour une meilleure circulation des flux, une approche orientée agents pour l'ordonnancement distribué des tâches de livraison qui permet d'apporter une solution à un problème de gestion de flux fortement distribué, et finalement un agent holonique dont le rôle est de fournir une estimation de la consommation future d'un site logistique en alliant l'expertise humaine aux modèles mathématiques.

La mise en œuvre et l'opérationnalisation des approches proposées, à l'aide d'un système multi-agent, montre que notre approche répond à notre problématique et apporte des solutions de bonnes qualités.

Les expérimentations réalisées sur la base de notre application, ont donné lieu à des propositions et à des réflexions. Il s'agit aussi bien d'aspects théoriques ou techniques de notre approche que de perspectives d'amélioration et d'extension.

Plusieurs perspectives sont d'ores et déjà positionnées dans notre planning de travail suivant les travaux de cette thèse. Parmi elles, l'enchainement du développement de l'outil pour améliorer la gestion des flux en concevant des interfaces conviviales et de meilleure qualité, permettant une facilité d'utilisation. Mais aussi l'intégration de certaines fonctionnalités complémentaires. Nous nous intéressons, dans ce sens, à mettre en œuvre des mécanismes d'anticipation, qui permettent de prévoir, détecter et éviter les situations non désirables dans la chaîne logistique (Kaddoussi, 2011-a).

Les approches utilisées dans ces travaux de recherche, notamment la gestion des chaînes logistiques et les systèmes multi-agents, sont souples et évolutifs. Une autre perspective consiste également à étendre nos résultats à d'autres secteurs, et notamment en logistique civile, grâce à la généricité de notre modèle.

Bibliographie

Artiba, A. (1990). *Contribution à la construction d'un système d'aide à la planification et à l'ordonnancement des lignes parallèles multi-produits.* Thèse de Doctorat. Université de Valenciennes et du Hainaut- Cambresis.

Barbat, B., Candea, C., Zamfirescu, C. (2001). Holons and agents in robotic teams: a synergistic approach. *In Proceedings of ENAI'2001*. pp. 654-660. ISBN 3-906454-25-8.

Barbier, F. et al., (2002). Composants dans l'ingénieurie des systèmes d'information : concepts clés et techniques de réutilisation, *actes des deuxièmes assises nationales du GdR I3 (Information - Interaction - Intelligence).*

Beamon, 1998 : Beamon, B.M., *Supply chain design and analysis : Models and methods*. International Journal of Production Economics, 55 (1998) 281-294. 1998.

Boissier, O. (2000). Visio-conférence et partages d'applications, rapport contrat d'objectif financé par la Région Rhône-Alpes. Technical report, ENSM.SE, March 2000.

Bond, A.H. & Gasser, L. (1988). *Readings in Distributed Artificial Intelligence*. San Mateo CA: Morgan-Kaufmann Publishers.

Botta-Genoulaz, 2005 : Botta-Genoulaz, V., *Principes et Méthodes pour l'Intégration et l'Optimisation du pilotage des Systèmes de Production et des Chaînes Logistiques*. Rapport de HDR, tome 1, Institut National des Sciences Appliquées de Lyon et Université Claude Bernard de Lyon 1, 2005.

Bradshaw, J. (1997). An introduction to software agents. (A. P. Press, Éd.) *Software agents*, pp. 3-46.

Briot, J.-P. (2009). Composants logiciels et systèmes multi-agents, *Technologies des systèmes multi-agents et applications industrielles*.: Hermès Lavoisier, ch. 5, pp. 147-187.

Bürckert, H., Fischer, K., and G.Vierke (1998). Transportation Scheduling with Holonic MAS - The TeleTruck Approach. *In Conf. on Practical Applications of Intelligent Agents and Multiagents*, pp. 577-590.

Burke, E., Hart, E., Kendall, G., Newall, J., Hart, E., Ross, P., and Schulenburg, S. (2003). Handbook of Meta-Heuristics, chapter Hyper-Heuristics : An Emerging Direction in Modern Search Technology, pp. 457–474. Kluwer Academic.

Cammarata, S., McArthur, D. and Steeb, R. (1988). Strategies of cooperation in distributed problem solving. *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 767-770, 1983. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser, editors, pages 102-105, Morgan Kaufmann, 1988).

Carlier, J. & Chrétienne, P. (1988). Les problèmes d'ordonnancement/ Modélisation/ Complexité/ Algorithmes. Edition Masson, Paris, France.

Chaib-draa, B. (1995). Industrial Applications of Distributed AI. *Communication of ACM*, 38(11). pp. 49-53.

Chaib-draa, B. (1994). Distributed Artificial Intelligence : An overview. In *Encyclopedia Of Computer Science And Technology*, Marcel Dekker Ed., vol. 31, pp. 215-243.

Chandy, K.M. & Misra, J. (1978). *A nontrivial example of concurrent processing distributed simulation*, Proceedings of IEEE COMPSAC'78, p.822-826.

Charles, A. (2010). *Improving the design and management of agile supply chains: feedback and application in the context of humanitarian aid.* Institut National Polytechnique, Université de Toulouse.

Christopher, M. (1992). *Logistics and supply chain management.* London: Pitman Publishing.

Colin, Mathé, & Tixié. (1981). *La logistique au service de l'entreprise.*

Cossentino, M. (2001). Different Perspectives in Designing Multi-Agent System, *Designing Multi-Agent System, AgeS'02 (Agent Technology and Software Engineering) Workshop at NodE'02*.

Courty, P. (2003). *Les enjeux industriels et les nouvelles problématiques scientifiques - De la logistique à la logistique globale.*

Croom, S., Romano, P., & Giannakis, M. (2000). Supply chain management: an Analytical framework for critical literature review. *European Journal of Purchasing & Supply Management* , Vol.6, pp. 67-83.

Daduna, J.R. and Paixão, J.M.P. "Vehicle scheduling for public mass transit – an overview," *Computer-Aided Transit Scheduling Notes de Lectures dans Economics and Mathematical System*, vol. 430, pp. 76–90, 1995.

Deguet, J. (2008). *Intégration de l'émergence au sein des systèmes multi-agents. Une étude appliquée à la recherche heuristique*. Thèse de l'Université Joseph Fourier, 2008.

DeLoach S. A. (2005). Engineering Organization-BasedMultiagent Systems. *SELMAS*, pp. 109-125.

Demazeau, Y. (1995). From interactions to collective behavior in agent-based systems. *In Proceedings of the 1st European conference on cognitive sciences*, 1995.

De Wolf, T. & Holvoet, T. (2005). *Emergence Versus Self-Organization: Different Concepts but Promising When Combined*. Engineering Self Organizing Systems: Methodologies and Applications, Lecture Notes in Computer Science, Volume 3464, pp. 1 – 15.

Doniec, A. (2006). *Prise en compte des comportements anticipatifs dans la coordination multi-agent : application à la simulation de trafic en carrefour*. Thèse de doctorat. France, Université de Valenciennes et du Hainaut Cambrésis.

Drogoul, A. (1993). *De La Simulation Multi-Agent A La Résolution Collective de Problèmes: Une Étude De l'Émergence De Structures D'Organisation Dans Les Systèmes Multi-Agents*. Thèse de Doctorat. Université Paris VI.

Drogoul, A. (2000). *Systèmes multi-agents situés*. Mémoire d'habilitation à diriger des recherches, Université Paris 6.

Ejday, M. (2011). *Optimisation Multi-Objectifs à base de Métamodèle pour les procédés de mise en forme.* Thèse de doctorat, École des MINES ParisTech, Paris.

Feki, M. F. (2010). *Optimisation distribuée pour la recherche des itinéraires multi-opérateurs dans un réseau de transport co-modal.* Thèse de doctorat, Ecole Centrale de Lille.

Ferber, J.(1995). *Les Systèmes Multi-Agents vers une intelligence collective.* InterEditions.

Fraser, A.S. (1957). Simulation of genetic systems by automatic digital computers. A*ustralian Journal of Biological Sciences*, 10:484_491.

Freling, R., Wagelmans, A.P.M. and Paixão, J.M.P. (1999) "An overview of models and techniques for integrating vehicle and crew scheduling," *Wilson, N.H.M. (Ed.) Computer-Aided Transit Scheduling. Notes de Lecture dans Economics and Mathematical Systems*, vol. 471, pp. 441– 460.

Ganeshan, R., Jack, E., Magazine, M., & Stephens, P. (1998). *A Taxonomic Review of Supply Chain Management Research, in Quantitative Models for Supply Chain Management.* (K. A. Publishers, Éd.) Boston.

Georgé, J.P., Gleizes, M.P., Glize, P. (2003). Conception de systèmes adaptatifs à fonctionnalité émergente : la théorie AMAS, *Revue des sciences et technologie de l'information*, vol.17, n° 4, p. 591-626, Hermès.

Ghanmi, A., Campbell, G.B., Gibbons, T.A. (2008). Modeling and simulation of multinational inter-theatre logistics distribution. *In Proceedings of the 40th Conference on Winter Simulation*. Pub. Winter Simulation Conference.

Glover, F. (1986). Future paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research*, 5:533-549.

Gomez Sanz J., Fuentes R. (2002). Agent Oriented System Engineering with INGENIAS. *Fourth Iberoamerican Workshop on Multi-Agent Systems, Iberagents'02.*

Gupta, M., Ko, H.-J. et Min, H. (2002). « TOC-based performance measures and five focusing steps in jobshop manufacturing environment ». *International Journal of Production Research*, 40(4), pp. 907-930.

Herrera, C. (2011). *Cadre générique de planification logistique dans un contexte de décisions centralisées et distribuées.* Thèse de doctorat, Université Henri Poincaré – Nancy 1.

Huang, G.Q., LAU, J.S.K. et Mak, K.L. (2003). *The impacts of sharing production information on supply chain dynamics : a review of the literature, International Journal of Production Research*, 2003, Vol. 41, No. 7, pp. 1483-1517.

Jadbabaie, A. , Lin, J. & Morese, A.S. (2003). Coordination of groups of mo-bile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 6(48): 988-1001.

Jamont, J-P. (2005). *DIAMOND : Une Approche pour la conception de Systèmes Multi-agents Embarqués*. Thèse de Doctorat, INPG, Septembre 2005.

Jeannin, M. (2008). *Négociation combinée et multicritères d'agents coopératifs.* Thèse de doctorat de l'université de Pierre et Marie Curie, Juillet 2008.

Jennings, N.R., Sycara, K. et Wooldridge, M. (1998). *A Roadmap of Agent Research and Development.* Autonomous Agents and Multi-Agent Systems, vol. 1, n° 1, pp. 7-38.

Jiang, J. & Jiang, Y. (2008). Convergence at Prominent Agents: A Non-Flat Synchronization Model of Situated Multi-Agents (Short Paper). *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Mueller and Parsons (eds.), Estoril, Portugal, pp.1493-1496.

Kaddoussi, A., Zoghlami, N., Zgaya, H., Hammadi, S. (2012-a). An agent-based distributed scheduling for a crisis management supply chain, The International Journal of Computational Intelligence Systems (IJCIS – Impact factor **1,471**), Vol. 6, N° 1, January 2013.

Kaddoussi, A., Zgaya, H., Hammadi, S., Duflos, E. & Vanheeghe, P. (2012-b). A need estimating agent-based tool for resources forecasting. T*he 16th World Multi-Conference on. Systemics, Cybernetics and Informatics*: *WMSCI 2012*. July 17th - 20th, 2012, *Orlando*, Florida, USA.

Kaddoussi, A., Zoghlami, Z., Zgaya, H., Hammadi, S. & Bretaudeau, F. (2011-a). A Preventive anticipation model for crisis management supply chain. *The 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC'11)*, 9-12 Octobre 2011, Anchorage, Alaska.

Kaddoussi, A., Zoghlami, Z., Zgaya, H., Hammadi. (2011-b) An Agent-based distributed Scheduling for Military Logistics, *The 11th International Conference on Intelligent Systems Design and Applications (ISDA'11)*, 22-24 Novembre 2011, Cordoba, Spain.

Kaddoussi, A., Zgaya, H., Hammadi, S. & Bretaudeau, F. (2011-c). Disruption Management Optimization for Military Logistics, *The 12th EANN/ 7th AIAI Joint Conferences : Engineering Applications of Neural Networks/ Artificial Intelligence Applications and Innovations*, Corfu, Grèce; Artificial Intelligence Applications and Innovations IFIP Advances in Information and Communication Technology. Volume 364, 2011, pp 61-66.

Kammoun, M. A. (2007). *Conception d'un système d'information pour l'aide au déplacement multimodal : Une approche multi-agents pour la recherche et la composition des itinéraires en ligne.* Lille: Ecole Centrale de Lille.

Koestler, A. (1968), The ghost in the machine, Macmillan.

Kouiss K., Pierreval, H. (1995). Systèmes multi-agents : directions actuelles pour les systèmes de production. *La productivité dans un monde sans frontières, Congrès International de Génie Industriel de Montreal*, Quebec, pp. 2029-2038.

Kubota F., S. Sato et M. Nakano, 1999. « Enterprise modelling and simulation platform integrating manufacturing system design and supply chain ». *Proceedings IEEE International Conference on Systems, Man, and Cybernetics*, 4, p. 511-515.

Krushal, J. (1956). *On the shortest spanning subtree of a graph and the traveling salesman problem*, Proc. Amer. Math. Soc., vol. 16. pp. 48-50.

Lee, Y.H., Kim, S.H. et Moon, C. (2002). Production distribution planning in supply chain using a hybrid approach . *Production Planning & Control*, 13(1), p.35-46.

Luder A., J. Peschke, T. Sauter, S. Deter et D. Diep, 2004. « Distributed Intelligence for Plant Automation Based on Multi-Agent Systems: The PABADIS Approach ». *Production Planning and Control*, 15(2), p. 201-212.

Lummus, R., Vokurka, R., & Albert, K. (1998). Strategic supply chain planning. *Production and Inventory Management Journal* , 39(3), 49-58.

Maria, A. (1997). *Introduction to Modeling and Simulation*, In Proceedings of the Winter Simulation Conference.

Maurer, P. M. (2000). Components: What if they gave a revolution and nobody came, *IEEE Software*, pp. 28-34.

Meignan, D. (2008). *Une approche organisationnelle et multi-agents pour la modélisation et l'implantation de métaheuristiques*, Thèse de Doctorat en Informatique, UTBM, décembre 2008.

Melzak, A. Z. (1961). *On the problem of Steiner*. s.l. : Math. Bull. pp. 143-150. Vol. 4

Mentzer, J., De Witt, W., Keebler, J., Min, S., Nix, N., Smith, C., et al. (2001). Defining supply chain management. *Journal of business logistics* , Vol.22, No.2.

Mertins K., Rabe, M. et Jakel, F.-W. (2005). *Distributed modeling and simulation of supply chains*, International Journal of Computer Integrated Manufacturing, 18(5), p. 342-349, 2005.

Metropolis N., Rosenbluth A., Rosenbluth M., Teller A. et Teller E. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, Vol. 21, 1953.

Nwana, H., Ndumu, D., Lee, L., & Collis, J. (1999). ZEUS: A tool-kit for building distributed multiagent systems. *Applied Artifical Intelligence, 13* , 129–186.

Oussalah, M. (2005). *Ingénierie des composants - Concepts, techniques et outils, Vuibert*. Vuibert.

Padgham L., Winikoff M. (2003). Prometheus: A Methodology for Developing Intelligent Agents, Giunchiglia F., Odell J., Weiß G. (dir.), *Agent-Oriented Software Engineering III, Third International Workshop, AOSE 2002*, Bologne, Italie, 15 juillet 2002, Revised Papers and Invited Contributions, vol. 2585 de *LNCS*, Springer, p. 174-185, 2003

Parunak, H. V. D., Brueckner, S., Fleischer, M., and Odell, J. (2003). A design taxonomy of multi-agent interactions. Lecture Notes in Computer Science, 2935(4) :123–137.

Perrot, J.-F. (1998). Objets, classes et héritage : Définitions, chapitre de Langages et modèles à objets - État des recherches et perspectives. *Collection Didactique*, INRIA, Ed., pp. 3–31.

Pham, 02: M.C. Pham. *La prise en charge du patient a l'hopital : mieux gerer la complexite de la coordination des acteurs.* Memoire de l'Ecole Nationale de la Sante Publique de Renne,

Poirier, C., & Reiter, S. (2001). *La Supply Chain - Optimiser la chaîne logistique et le réseau interentreprises.* Dunod.

Rao, A. S. et Georgeff, M. (1995). BDI Agents : from theory to practice. In the *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95).* pp. 312-319, San Francisco, CA, USA.

Reynolds, C. W. (1987). Flocks herds, and schools: a distributed behavioral model. Computer Graphics, 4(21): 25-34.

Rohde, J., Meyr, H. & Wagner, M. (2000). *Die supply chain planning matrix,* in : PPS Management, Vol. 5, No.1, Berlin, pp. 10-15.

Rota-Franz, K., Thierry, C., & Be, G. (2001). *Gestion des Flux dans les Chaînes Logistiques: Performances industrielles et gestion des flux.* Hermès Traité IC2.

Rousseau, J-M. (1985). *Computer Scheduling of Public Transport*. North-Holland, vol. II.

Rouillard, J., Vantroys, T. and Chevrin, V. (2007). *Architecture Orientée Services, Une approche pragmatique des SOA*, Editions Vuiberts, Ed.

Sghaier, M. (2011). *Combinaison des techniques d'optimisation et de l'intelligence artificielle distribuée pour la mise en place d'un système de covoiturage dynamique*. Ecole Centrale de Lille.

Shalizi, C. (2001). *Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata*. Thèse de Doctorat. University of Wisconsin, Madison, USA.

Shlaer, S. & Mellor, S. (1988). *Object oriented System Analysis : modeling the world in data.* Prentice Hall.

Shen, W., Hao, Q., Yoon, H. J. and Norrie, D. H. (2006). Applications of agent-based systems in intelligent manufacturing: An updated review, Advanced Engineering Informatics, 20(4), 415-431, 2006.

Siarry, P. (2002). Application des métaheuristiques d'optimisation en Electronique, Techniques de l'Ingénieur. vol. 9.

Simchi-Levi, D., & Kaminsky, P. (2003). *Designing and Managing the Supply Chain: Concepts, Strategies and Case Studies.* Boston, MA.: McGraw Hill Professional.

Strugeon, E. L. (1995). *Une méthodologie d'auto-adaptation d'un système multi-agents cognitifs.* Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, France.

Supply Chain Council. (s.d.). Récupéré sur Supply Chain Council: http://supply-chain.org/

Takahashi, H. & Matsuyama, A. (1980). An Approximate Solution for the Steiner P Problem in Graphs", *Math. Japonica* 24(6), pp, 573-577.

Talbi, E.-G. & Bachelet, V. (2004). Cosearch : A parallel co-evolutionary metaheuristic. I*n Int. workshop on hybrid metaheuritics*, pp. 127–140.

Thomas et Griffin, 1996 : D.J. Thomas, P.M. Griffin. *Coordinated supply chain management*. European Journal of Operational Research. 94, 1996, pp 1-15.

Toussaint, H. (2010). *Algorithmique rapide pour les problèmes de tournées et d'ordonnancement*. Thèse de Doctorat, Université Blaise Pascal - Clermont-Ferrand II.

Turner S.J., W. Cai et B.P. Gan,*Adapting a supply-chain simulation for HLA*, 4th International Workshop on Distributed Simulation and Real-Time Applications, p. 71-78, 2000.

Ulieru, M. and Geras, A. (2002). Emergent holarchies for e-health applications : a case in glaucoma diagnosis. I*n IEEE IECON'02*, volume 4, pages 2957-2961.

Vakharia, A. (2002). e-Business and supply chain management. *A journal of the Decision Sciences Institute* , Vol. 33, Issue 4, pp. 495-504.

Vallin, M. et Coello, J.M.A. (2003). An agent for web information dissemination based on a genetic algorithm. I*n IEEE International Conference on Systems, Man and Cybernetics - IEEE SMC'03*, pp. 3834–3839, Hyatt Regency, Washington, D.C., USA, 5 - 8 October 2003. IEEE Press.

Vincent et al., 2004 : L. Vincent, G. Neubert, D. Llerena et C. Pellegrin. *Synthèse des approches SCM existantes (livrable 4.1)* in *Annexe 5 du rapport de 1ère année du projet COPILOTES*, 47-73 (2004).

Vauvert, G. et El Fallah - Seghrouchni, A. (2000). Démonstration : Formation de coalitions pour des agents libres ; application à la gestion de transport aérien. *In Proceedings de JFIAD-SMA'2000*. 8èmes Journées Francophones de l'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents. - Saint-Jean-La-Vêtre, France, 2-4 Octobre 2000.

Warnecke, H. (1993). The Fractal Company.

Weiss, G. (2000). *Multiagent Systems, A Modern Approach to distributed Artificial Intelligence*. The MIT Press.

Wirfs-Brock, R., Wilkerson, B. & Wiener, L. (1990). *Designing Object-Oriented Software.*Prentice Hall.

Wooldridge, M. et Jennings, N. (1995). *Intelligent Agent: Theory and practice, Knowledge Engineering Review*, vol. 10, n° 2, pp. 115-142.

Zelikovsky, A.Z. (1993). *A 11/6-Approximation Algorithm for the Network Steiner Problem.* Algorithmica, vol. 9. pp. 463-470.

Zgaya, H. (2007). *Conception et optimisation distribuée d'un système d'information d'aide à la mobilité urbaine : Une approche multi-agent pour la recherche et la composition des services liés au transport.* Thèse de doctorat, Ecole Centrale de Lille, Lille.

Zoghlami, N. (2008). *Optimisation à base d'agents communicants des flux logistiques pour la gestion de crises.* Thèse de Doctorat. Ecole Centrale de Lille.

Optimisation des flux logistiques : vers une gestion avancée de la situation de crise

Résumé

La logistique de gestion de crise fait de plus en plus parler d'elle. En effet, que ce soit au sujet de conflits géopolitiques ou de catastrophes naturelles, ou encore lorsqu'il s'agit d'urgences de proximité, on ne peut plus passer à côté de la gestion de crise pour faire face à des faits internationaux ou nationaux. La logistique de gestion de crise œuvre pour que les secours soient optimisés et que l'aide et le ravitaillement en nourriture, eau, médicaments, soient apportés rapidement aux zones sinistrées. Les travaux de recherche présentés dans cette thèse, dans le cadre d'une coopération avec le département logistique d'EADS, visent à apporter une réponse à ces défis. Nous présentons, dans ce contexte, des solutions basées sur une approche de modélisation, optimisation et simulation orientée agents de flux logistiques. Les problèmes de gestion de flux abordés nous ont permis d'étudier et développer trois volets d'investigation : un module d'aide au positionnement des zones logistiques qui permet de minimiser les distances entre les zones et d'optimiser leurs nombre et ce pour une meilleure circulation des flux, une approche orientée agents pour l'ordonnancement distribué des tâches de livraison qui permet d'apporter une solution à un problème de gestion des flux fortement distribué, et finalement un agent holonique qui permet de fournir une estimation de la consommation à venir d'un site logistique, en alliant l'expertise des logisticiens aux modèles à base de logique floue. Pour démontrer l'efficacité des approches utilisées, un démonstrateur baptisé OBAC (**O**ptimisation à **B**ase d'**A**gents **C**ommunicants) a été réalisé, intégrant toutes ces approches et applications dédiées à la gestion de crise.

Mots clés : Gestion de crise, Système multi-agent, Logistique de gestion de crise, Optimisation, Ordonnancement distribué.

Optimization of logistics flows: towards an advanced crisis management supply chain

Abstract

Nowadays, crisis management logistics is facing new challenges. Indeed, geopolitical conflicts, natural disasters or emergencies can cause big damages and so require rapid and effective response. Due to their sudden occurrence, we are countering a high disturbances context: a Crisis Management Supply Chain (CMSC). Working in such an uncertain environment incites to be equipped with optimization and cooperation mechanisms assuring all the chain actors satisfaction, while acting in a collective way to reach a common objective: the crisis management.

In this thesis, we focus on the definition of a modeling approach and an agent-based simulation of the Crisis Management Supply Chain. We propose a decision support system that deals with three problems: the optimal positioning of logistics zones to facilitate the flows circulation, an innovative method for solving a highly-distributed delivery scheduling problem, based on a multi-agent system, for the distribution of relief materials (food, water, clothes, etc.) to the areas affected by the disaster, and finally a need estimating agent to give an accurate forecast of resources' consumption in the logistic zones. Blending the agent paradigm with the optimization technics helped reach our goals of implementing a large-scale decision support system. The simulation results highlight our contributions.

Keywords: Crisis management, Multi-agent systems, Logistics, Optimization, Distributed scheduling.