# Conception et optimisation d'un système d'information d'aide à la mobilité : une approche multi-agent pour la recherche et la composition des services dans un espace ubiquitaire

Ayoub Bousselmi

**N° d'ordre : 266**

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


## Ayoub BOUSSELMI


Doctorat Delivre par l'Ecole Centrale de Lille

# Conception et optimisation d'un système d'information d'aide à la mobilité: une approche multi-agent pour la recherche et la composition des services dans un espace ubiquitaire

Soutenue publiquement le 04 Juin 2015 devant le jury d'examen :

| | |
|---|---|
| **Président :** | Sophie Dupuy-Chessa, Professeur, Laboratoire d'Informatique de Grenoble |
| **Rapporteur :** | Alain Quilliot, Professeur, Université Blaise Pascal, Clermont-Ferrand |
| **Rapporteur :** | Aziz Moukrim, Professeur, Université de Technologie de Compiègne |
| **Examinateur :** | Nesrine Zoghlami, MCF, Ecole Nationale d'Ingénieurs de Tunis |
| **Examinateur :** | Hayfa Zgaya, MCU HDR, ILIS, Université Lille 2 |
| **Examinateur :** | Thomas Bourdeaud'huy, MCU, Ecole Centrale de Lille |
| **Directeur :** | Slim Hammadi, Professeur, Ecole Centrale de Lille |
| **Co-Directeur :** | Christian Tahon, Professeur, Université de Valenciennes |


Thèse préparée dans le laboratoire CRIStAL UMR CNRS 9189 à l'Ecole Centrale de Lille
École Doctorale des Sciences Pour l'Ingénieur Lille Nord de France – 072
**PRES Université Lilleeee Nord-de-France**

*“Se donner du mal pour les petites choses, c’est parvenir aux grandes, avec le temps”*

Samuel Beckett

# Résumé étendu

De nos jours, les technologies d'information et de communication représentent un atout majeur pour les systèmes d'information d'aide au déplacement dans différents domaines, en particulier dans le domaine du transport. De plus, dans un contexte ubiquitaire, les objets connectés (Smartphones, tablettes, etc.) sont capables d'interagir avec les utilisateurs pour leurs fournir des services innovants et les aider à optimiser leurs plans de déplacement. En effet, le nombre d'utilisateurs ainsi que le nombre de fournisseurs des services demandés par ces utilisateurs sont en pleine augmentation. Cette croissance implique un aspect de concurrence et nécessite des choix optimisés. Dans ce cadre, l'objectif de cette thèse est de concevoir et développer un système d'aide au déplacement qui couvre non seulement les services de déplacements quotidiens mais aussi les services touristiques, culturels et bien d'autres.

Ordinairement, les fournisseurs des services de transport proposent des solutions centralisées, telles que les portails web, pour la réservation des moyens de transport et la présentation des horaires, des itinéraires et parfois de quelques évènements culturels. Cependant, ces types de systèmes risquent de ne pas être en mesure de satisfaire les demandes croissantes en termes de services d'aide au déplacement. En effet, l'introduction du paradigme *Informatique Ubiquitaire*, appelée aussi *Internet des Objets*, permet de proposer des nouvelles solutions pour les futurs systèmes d'informations. Néanmoins, la naissance d'objets intelligents capables d'offrir des services et des terminaux mobiles capables de communiquer n'importe où et n'importe quand, implique de véritables contraintes : les charges élevées du réseau, la topologie distribuée, l'environnement dynamique et hétérogène, etc.

Les travaux de recherche présentés dans ce manuscrit proposent la mise en place d'une Plateforme de Recherche et de composition des Services d'Aide à la Mobilité (PRoSAM) permettant d'optimiser les tâches de recherche, de composition et de distribution des Informations de Mobilité Avancée (IMA). D'une part, les fournisseurs de service appartenant au Réseau Distribué de Transport Comodal (RDTC) peuvent proposer les mêmes IMA à différents coûts, délais de réponse, quantités de données, etc. Dans ce cas, le système doit nécessairement délivrer aux utilisateurs des IMA optimisées. D'autre part, les utilisateurs demandent les services en composant des requêtes qui peuvent être nombreuses et simultanées. Dans ce cas, le système doit être aussi en mesure de gérer ce flux d'information tout en minimisant les délais de transmission et en préservant sa capacité de robustesse pour faire face à la montée en charge.

L'aspect dynamique et distribué du problème nous a conduit à adopter une modélisation orientée agent afin de s'adapter aux conditions d'un environnement ubiquitaire. Le système d'agents proposé comporte deux sous-systèmes principaux : le connecteur et le voisinage collaboratif. En s'appuyant sur des métaheuristiques pour la recherche et la localisation des IMA, un connecteur emploie plusieurs types d'agents pour traiter les requêtes simultanées, récupérer les données et générer les réponses. Quant à un voisinage collaboratif, son rôle est d'assister le connecteur pour la distribution des services

afin de minimiser les délais de transmission. En effet, grâce à une approche dynamique de changement de rôles des agents représentant les utilisateurs et un protocole de négociation innovant, les clients sont capables d'échanger les services d'une manière autonome et d'établir des accords totaux ou partiels en fonction de l'offre et de la demande.

Enfin, les résultats de simulation présentés dans cette thèse démontrent l'efficacité des approches proposées et notamment l'impact du voisinage collaboratif sur les performances du système ubiquitaire. D'une part, la comparaison menée entre deux systèmes, sans et avec le changement dynamique des rôles, prouve la fiabilité de cette stratégie. D'autre part, la simulation du protocole de négociation à accord-partiel proposé montre sa capacité à réduire le flux de communication et améliorer la capacité globale du système à gérer la charge du réseau.

# Remerciements

Je tiens à remercier en tout premier lieu mon directeur de thèse le Professeur Slim Hammadi, Professeur à l'Ecole Centrale de Lille, de m'avoir accueilli, soutenu, encouragé et encadré tout au long de cette thèse, de m'avoir permis de travailler sur un sujet passionnant et innovant, d'avoir cru en mes capacités et de m'avoir fait découvrir réellement et d'aimer sincèrement la recherche.

Je remercie également le Professeur Christian Tahon, co-directeur de cette thèse du laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines de l'Université de Valenciennes et du Hainaut-Cambrésis, et sans qui ce projet n'aurait peut-être pas abouti.

Je remercie tout aussi chaleureusement mon premier encadrant et examinateur de cette thèse le Docteur Hayfa Zgaya, maître de conférences et HDR à la faculté d'Ingénierie et Management de la Santé à l'Université Lille 2 pour sa présence, son encouragement, sa perspicacité et son énorme soutien scientifique.

Je tiens également à remercier sincèrement mon deuxième encadrant et examinateur de cette thèse le Docteur Thomas Bourdeaud'huy, maître de conférences à l'Ecole Centrale de Lille, pour m'avoir soutenue continuellement et inlassablement. Il a fait preuve d'une présence scientifique imposante pour voir ces travaux prendre forme progressivement et enfin aboutir.

Mes remerciements s'adressent exceptionnellement au Professeur Sofie Dupuy-Chessa, Professeur de l'Université Pierre Mendès-France Grenoble 2, de m'avoir honoré en acceptant de présider mon jury de soutenance.

Mes plus sincères remerciements s'adressent au Professeur Alain Quilliot, Directeur du LIMOS et de la fédaration CNRS TIMS et Professeur d'Informatique à l'Université Blaise Pascal de Clermont-Ferrand et au Professeur Aziz Moukrim, Professeur à l'Université de Technologie de Compiègne, 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.

J'adresse des remerciements tout à fait particuliers au Docteur Nesrine Zoghlami, maître de conférences à l'Ecole Nationale d'Ingénieurs de Tunis pour avoir accepté d'être parmi les examinateurs de cette thèse.

Je remercie aussi le personnel du CRIStAL et de l'Ecole Centrale de Lille, en particulier le personel de la bibliothèque, pour leur gentillesse, disponibilité et serviabilité. Enfin, je remercie du plus profond de mon cœur ma famille qui m'a toujours encouragée et soutenue et à qui je dédie ce travail.

# Table des matières

# Table des figures

# Liste des tableaux

# Liste des algorithmes

# Abréviations

| | |
|---|---|
| **Ac** | **A**gent **c**omposeur |
| **Ad** | **A**gent **d**écomposeur |
| **AFS** | **A**nnuaire des **F**ournisseurs de **S**ervices |
| **Ao** | **A**gent **o**ptimisateur |
| **Ar** | **A**gent **r**amasseur |
| **Au** | **A**gent **u**tilisateur |
| **CC** | **C**loud **C**omputing (informatique des nuages) |
| **ETD** | **E**ntrepôt **T**emporaire de **D**onnées |
| **GC** | **G**rid **C**omputing (grilles informatiques) |
| **GPS** | **G**lobal **P**ositioning **S**ystem |
| **HTTP** | **H**yper**T**ext **T**ransfer **P**rotocol (protocole de transfert hypertexte) |
| **IA** | **I**ntelligence **A**rtificielle |
| **IAD** | **I**ntelligence **A**rtificielle **D**istribuée |
| **IAm** | **I**ntelligence **Am**biante |
| **IdO** | **I**nternet **d**es **O**bjets |
| **IMA** | **I**nformation de **M**obilité **A**vancée |
| **MC** | **M**obile **C**omputing (informatique mobile) |
| **MCC** | **M**obile **C**loud **C**omputing |
| **NFC** | **N**ear **F**ield **C**ommunication (communication en champ proche) |
| **P2P** | **P**air-à-**P**air ou système Pair-à-Pair |
| **PDA** | **P**ersonal **D**igital **A**ssitant (assistant numérique personnel) |
| **POMO** | **P**roblème d'**O**ptimisation **M**ulti-**O**bjectif |
| **PRoSAM** | **P**lateforme de **R**echerche et de c**o**mposition des **S**ervices d'**A**ide à la **M**obilité |
| **QoS** | **Q**uality **o**f **S**ervice (Qualité de Service) |
| **RDTC** | **R**éseau **D**istribué de **T**ransport **C**omodal |
| **REST** | **RE**presentational **S**tate **T**ransfer |

| | |
|---|---|
| **RFID** | **R**adio **F**requency **ID**entification |
| **RTS** | **R**essource de **T**raitement et de **S**tockage |
| **SI** | **S**ystèmes d'**I**nformation |
| **SIAM** | **S**ystème d'**I**nformation d'**A**ide à la **M**obilité |
| **SMA** | **S**ystème **M**ulti-**A**gent |
| **TIC** | **T**echnologies de l'**I**nformation et de la **C**ommunication |
| **TM** | **T**erminal **M**obile |
| **TRL** | **T**erminal à **R**essources **L**imitées |
| **Ubicomp** | **Ubi**quitous **comp**uting (informatique ubiquitaire) |
| **URI** | **U**niform **R**esource **I**dentifier |
| **WSN** | **W**ireless **S**ensor **N**etwork (réseau de capteurs sans-fil) |

*À ma chère famille...*

# Introduction Générale

## Contexte et Problématique

Le domaine du transport progresse continuellement, de l'intermodalité qui consiste à combiner successivement plusieurs modes de transport pour se déplacer, vers la multimodalité où différents modes de transport sont en concurrence et permettent un déplacement plus flexible, à la comodalité qui vise la complémentarité entre les différents modes de transport (publics et privés), et ce dans le but d'atteindre l'utilisation optimale et durable des ressources. La nouvelle forme d'évolution du transport est l'intégration des services d'aide à la mobilité urbaine. Un voyageur (client) a besoin d'être assisté avant et au cours de son voyage. Chaque client peut formuler un ensemble de requêtes de services qui vont être traitées par un système d'information d'aide à la mobilité (SIAM). Le rôle d'un tel système est de fournir les réponses nécessaires à ces demandes tout en optimisant le temps de recherche, le coût et la qualité des services recherchés. Un SIAM innovant ne doit plus être limité à fournir uniquement des informations de transport telles que les horaires des moyens de transport et les informations de sécurité des voyageurs ; ces derniers ont de plus en plus besoin d'être accompagnés par tout un ensemble de services capables de faciliter leur mobilité tout en préservant leur sécurité, de réduire leur temps perdu à la recherche d'informations pertinentes. Enfin, un SIAM doit exploiter efficacement les ressources fournies par les espaces communicants de demain : les espaces ubiquitaires.

Les informations qui assistent les voyageurs dans leurs déplacements sont récupérées à partir d'une couche servicielle émergeante qui fait le lien entre le client et son environnement ubiquitaire et permet une interaction bidirectionnelle en temps réel. En effet, les avancées technologiques au niveau de la miniaturisation de l'électronique ont permis la création de dispositifs électroniques et informatiques à différentes capacités de calcul et modes d'affichage pour devenir des éléments indispensables au quotidien. Ces dispositifs sont capables de communiquer pour fournir des services et augmenter notre environnement d'informations et de connaissances. Dans ce contexte, trois défis sont à relever. Le premier est sans doute la gestion de cette quantité débordante de services disponibles, chacun avec des coût, QoS et temps de réponse différents. Le deuxième défi est la topologie du réseau informatique distribué où les fournisseurs de services ne sont pas accessibles à travers une entité centrale. Dans ce cas, la solution apportée doit prendre en compte le caractère distribué et la dynamique élevée

de l'environnement. Le troisième défi est l'utilisation croissante des terminaux mobiles (Smartphones, Tablettes, Laptops, etc.) qui possèdent des ressources de stockage, de calcul et une autonomie limitées.

Certains services sont de plus en plus demandés, par exemple les services de calcul d'itinéraires offerts par les systèmes de géolocalisation, les services de musique et vidéo sur demande ou les services de partage des ressources. Ces services permettent de soutenir les voyageurs dans leurs déplacements en leur proposant une nouvelle expérience de mobilité intelligente. Ces services sont donc au cœur de la mobilité permettant de rendre les déplacements des voyageurs plus efficaces et agréables. En effet, les services de mobilité aident les utilisateurs à décider de leurs plans en leur fournissant en temps réel des solutions de déplacement optimisées en termes de coût et de nombre de changement de modes.

Cette thèse de doctorat est co-financée par l'ADEME [1] et la région Nord-pas-de-Calais [2]. Elle s'inscrit au sein du programme de recherche intitulé *villes et territoires durables* [3] de la stratégie RDI 2014-2020 de l'ADEME qui encourage les recherches autour de la transition énergétique et écologique. Ces travaux s'insèrent aussi dans le cadre du projet CISIT [4] sous l'objectif 1 : *gestion optimale des chaînes multimodales* [5]. Cette thèse est préparée au sein de l'équipe OSL (Optimisation des Systèmes Logistiques) du laboratoire CRIStAL. Notre équipe s'investit dans la conception et le développement des systèmes d'aide à la décision en combinant les outils d'optimisation distribuée avec le concept d'agent dans le but d'améliorer, sécuriser, et optimiser ces systèmes. Ces derniers sont adaptés pour des environnements de nature distribuée tels que : la chaîne logistique globale du transport comodal, les réseaux de soins, les réseaux de production multi sites et la gestion de crise multi zones. Les travaux présentés dans cette thèse se positionnent dans le thème de conception et d'optimisation des systèmes d'information d'aide au déplacement dans un contexte d'espace ubiquitaire.

# Objectifs et Contributions

Ce travail se situe à l'intersection des trois domaines suivants : l'informatique ubiquitaire, les systèmes multi-agents et l'optimisation distribuée. Le but de cette thèse est la modélisation et le développement d'un système d'information capable d'optimiser la recherche et la composition des services d'aide à la mobilité dans un environnement ubiquitaire. Nous baptisons le système proposé : "Plateforme de Recherche et de composition de Services d'Aide à la Mobilité (PRoSAM)". Ce système est capable de contourner les limites et les verrous des Systèmes d'Information (SI)

1. Agence de l'Environnement et de la Maîtrise de l'Énergie (http://www.ademe.fr/)
2. http://www.nordpasdecalais.fr/jcms/c_5001/accueil
3. Rapport technique: http://www.ademe.fr/.../ademe_orientations_recherche_dev_innov_v13_md.pdf
4. International Campus on Safety and Intermodality in Transportation (http://www.cisit.org/)
5. Rapport technique: http://www.cisit.org/.../RA_CISIT_2012_FR.pdf

actuels : nombre limité d'utilisateurs simultanés, résultats aléatoires non optimisés, résultats parfois limités et non compétitifs, recherche de services non automatisée, etc. Un état de l'art sur ces systèmes est présenté dans ce rapport. Ces derniers sont principalement basés sur des architectures centralisées limitant l'autonomie des fournisseurs et l'interopérabilité et la coopération avec des nouveaux partenaires avec un coût élevé de mise à jour des bases de données. De plus, plusieurs systèmes de recherche et de composition de services d'aide à la mobilité ont été proposés dans la littérature [Zidi, 2006, Zgaya, 2007, Feki, 2010, Sghaier, 2011]. Les solutions innovantes proposées dans ce contexte sont basées sur une modélisation distribuée orientée agent mais présentent des limitations comparables à celles des systèmes classiques.

Notre défi dans cette thèse est de créer un système d'information capable de s'introduire entre les fournisseurs de services dans un Réseau Distribué de Transport Comodal (RDTC) et le voyageur afin d'améliorer et optimiser aux services d'accompagnement de la mobilité. Le système proposé permet d'offrir non seulement des services d'aide à la mobilité mais aussi des services connexes d'une manière efficace, fiable et en temps réel à un grand nombre d'utilisateurs simultanément.

La contribution de ce travail est répartie sur trois niveaux. Le premier niveau est l'intégration des ressources clients dans le cycle de composition et d'optimisation des services d'aide à la mobilité. Le deuxième niveau est la proposition d'un protocole de négociation à accords partiels capable de réaliser des communications directes et simples entre les clients afin de permettre des actions de "vente" et "d'achat" des services. Le troisième niveau est la plateforme d'expérimentation numérique développée permettant de conduire différents scénarios de simulation qui valident nos approches.

## Plateforme de Recherche et de compostion des Services d'Aide à la Mobilité (PRoSAM)

La plateforme PRoSAM se base sur des travaux antérieurs [Zgaya, 2005, Zgaya et Hammadi, 2006, Zgaya et Hammadi, 2007, Zgaya, 2007] effectués au sein de notre équipe. PRoSAM représente une amélioration de ces travaux et propose un nouveau concept de modélisation des systèmes d'information basé sur une approche multi-agents orientée "rôle". Cette plateforme permet aux utilisateurs des moyens de transport de demander des services d'aide au déplacement. Ces services sont proposés par un ensemble de fournisseurs de services situés dans un Réseau Distribué de Transport Comodal (RDTC). Ces fournisseurs peuvent proposer les mêmes services à différents coûts, durées de traitement et quantités de données transférées.

Afin d'offrir les meilleurs services aux clients, PRoSAM dispose d'un noyau d'optimisation appelé *Connecteur*. Un connecteur est composé d'un ensemble d'agents qui coopèrent et coordonnent leurs actions afin de générer des solutions optimisées en se basant sur une approche évolutionnaire. Le connecteur se base aussi sur un paradigme particulier de la technologie agent : l'agent mobile. En effet, un agent mobile est capable de se déplacer d'un nœud à un autre dans un réseau distribué et exécuter des tâches localement sur chaque nœud visité. Le processus d'optimisation de la recherche des service génère dans ce cas des itinéraires pour les agents mobiles qui se déplacent selon ces itinéraires dans le RDTC afin de collecter les données demandées.

PRoSAM dipose aussi d'un composant appelé *voisinage collaboratif*. Ce dernier est composé d'un ensemble d'agents utilisateurs. Au sein d'un voisinage collaboratif, les utilisateurs sont capables d'échanger des services grâce à une approche orientée rôle. Cette approche s'inspire des système pair-à-pair. L'idée est d'utiliser les ressources de traitement et de stockage des terminaux mobiles des utilisateurs afin d'héberger des services au profit des membre du voisinage pour avoir un meilleur temps de réponse et mieux gérer la charge du réseau. Les agents utilisateurs sont capables d'avoir deux rôles : un rôle passif ou actif. Le rôle passif ne permet au client que de demander des services. Par contre, un rôle actif permet au client à la fois de demander des services et fournir les services qu'il vient de recevoir.

Le changement de rôle passif-actif et actif-passif est géré par le connecteur en utilisant une politique basée sur des permissions et des dates de validités des services. L'échange des services entre les utilisateurs est optimisé grâce à un protocole de négociation à accord partiel.

# Organisation du Mémoire

**Chapitre 1 : Aide à la Mobilité Urbaine : Vers une Mobilité Intelligente Avancée**. Ce chapitre introduit le contexte de cette thèse en terme d'aide à la mobilité. Il présente en particulier plusieurs systèmes d'information existants dans le domaine du transport et identifie les points forts et les points faibles de ces systèmes. Il présente ensuite la mobilité dans le contexte des espaces ubiquitaires et des technologies d'information et de communication. La notion d'information de mobilité avancée est ainsi introduite dans ce chapitre.

**Chapitre 2 : Modélisation et Optimisation des Systèmes d'Information au Profit de la Mobilité Avancée**. Ce chapitre présente une revue de littérature des systèmes d'information étendus sur deux plans. Le premier plan décrit les différentes architectures de traitement des informations

distribuées (i.e. distribution géographique et logique). Ceci regroupe les systèmes pair-à-pair, les grilles informatiques “Grid Computing” et l'informatique dans les nuages “Cloud Computding”. Le deuxième plan présente les propriétés et les caractéristiques des systèmes multi-agents et introduit l'optimisation évolutionnaire qui sera combinée avec ces derniers.

**Chapitre 3 : Proposition d'un Système d'Agents Optimisateurs pour la Recherche et la Distribution des Informations de Mobilité Avancée**. Ce chapitre présente le problème traité dans le cadre de cette thèse avant d'exposer le système d'information d'aide à la mobilité proposé basé sur le paradigme agent. Les approches d'optimisation de la recherche des informations de mobilité sont aussi présentées. En particulier, les algorithmes d'optimisation des itinéraires des agents mobiles sont proposés dans ce chapitre.

**Chapitre 4 : Protocole de Négociation pour Optimiser les Communications Agents du Système Proposé**. Ce chapitre présente la description dynamique du système proposé. Ce système est basé sur le changement de “rôle” de ses utilisateurs pouvant se transformer en fournisseurs de services. En particulier, l'algorithme d'administration du changement dynamique du rôle est présenté avec des exemples d'application. Ensuite, nous introduisons un protocole de communication à accords partiels nécessaire pour l'optimisation du processus d'échange des services de mobilité entre les utilisateurs du système.

**Chapitre 5 : Simulation et Analyse des Résultats**. Dans ce chapitre, nous présentons la conception, l'implémentation et le paramétrage d'un démonstrateur. Nous présentons ainsi quelques scénarios simulant la mise en application de la solution proposée dans le cadre de cette thèse. Nous présentons aussi les outils de développement utilisés ainsi qu'une analyse détaillée des résultats obtenus permettant d'évaluer l'efficacité des approches proposées.

**Conclusion Générale**. Ce rapport se termine par la présentation d'un bilan sur l'état actuel de nos travaux et sur les perspectives envisagées.

# Chapitre I

# Aide à la Mobilité Urbaine : Vers une Mobilité Intelligente Avancée

## I.1 Introduction

Ce chapitre présente en trois parties le contexte général de cette thèse. Dans la première partie, nous introduisons les atouts de l'aide à la mobilité urbaine ainsi que la notion d'information de mobilité avancée en tant que politique nationale et internationale permettant de mieux exploiter les ressources disponibles, respecter l'environnement et assurer une meilleure qualité de service.

Dans la deuxième partie, nous présentons quelques exemples de systèmes d'information conventionnels dans le domaine du transport, leurs spécifications, leurs avantages et leurs limitations.

Dans la troisième partie, nous introduisons les paradigmes de l'informatique ubiquitaire et de l'intelligence ambiante. Ces concepts représentent la base des systèmes d'information de mobilité de demain.

## I.2 Aide à la Mobilité Avancée

### I.2.1 Mobilité Urbaine

La mobilité urbaine représente l'ensemble des déplacements quotidiens effectués sur un espace urbain par différents modes de transport et pour différents motifs, d'une origine vers une destination.

Depuis les années quatre-vingt-dix, le déplacement interne des voyageurs en France augmente progressivement [1]. Les voyageurs, de plus en plus nombreux (Fig. I.1 [2]), parcourent quotidiennement et en interne (i.e. des parcours reliant un point de départ et un point d'arrivée situés à l'intérieur du pays) une distance de plus en plus longue (Fig. I.2). En outre, le volume de déplacements intérieurs selon les différents modes de transport ne cesse d'augmenter, comme le montre la Fig. I.3 [3] fournie par l'INSEE [4].

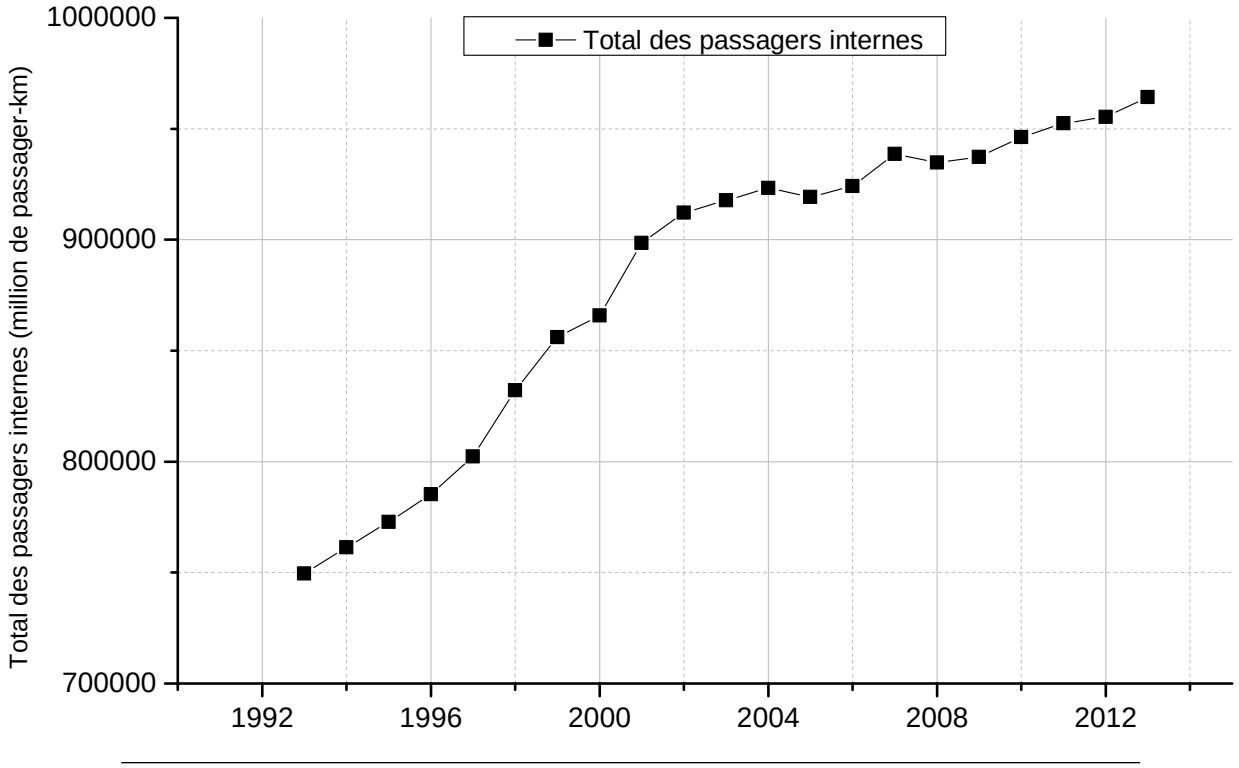



FIGURE I.1: Évolution du nombre de voyageurs et des déplacements intérieurs de 1990 à 2013. Source : OECD

Du point de vue de la demande, la mobilité est plus dispersée sur les plans de la géographie, des modes, des fréquences et des destinations. La mobilité domicile-travail n'est plus dominante et les Technologies de l'Information et de la Communication (TIC) (§I.2.3) enrichissent les déplacements par des informations appelées informations de mobilité avancée (IMA) grâce à leurs multiples usages pendant les temps de transport. Cette nouvelle mobilité s'enrichit donc d'activités conjointes aux déplacements.

1. http://www.internationaltransportforum.org/ et http://www.insee.fr/fr/
2. Organisation for Economic Co-operation and Development (http://www.oecd.org/)
3. Indice base 100 : l'indice d'une grandeur est le rapport entre la valeur de cette grandeur au cours d'une période courante et sa valeur au cours d'une période de base. Il mesure la variation relative de la valeur entre la période de base et la période courante. Souvent, on multiplie le rapport par 100 ; on dit : indice base 100 à telle période. Les indices permettent de calculer et de comparer facilement les évolutions de plusieurs grandeurs entre deux périodes données.
4. Institut National de la Statistique et des Études Économiques (http://www.insee.fr/fr/), Service de l'Observation et des Statistiques

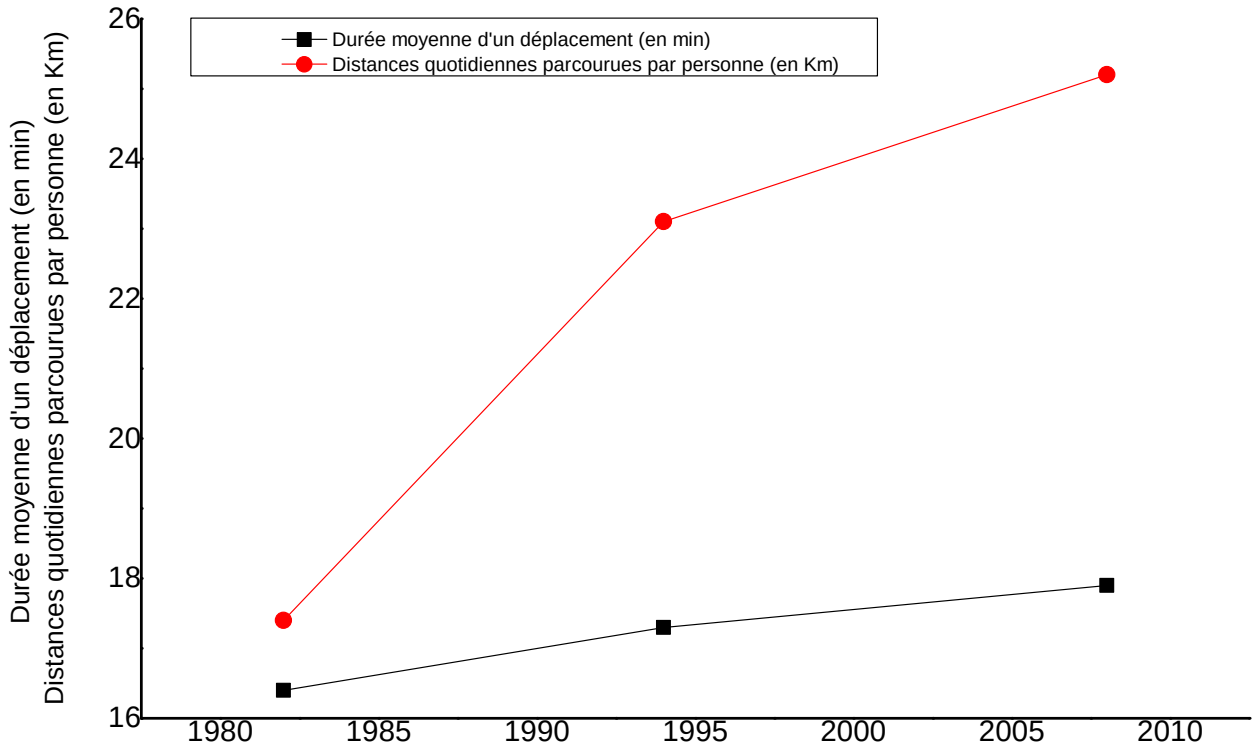



FIGURE I.2: Distances parcourues et durées des déplacements locaux, entre 1982 et 2008 [Armoogum et al., 2010]

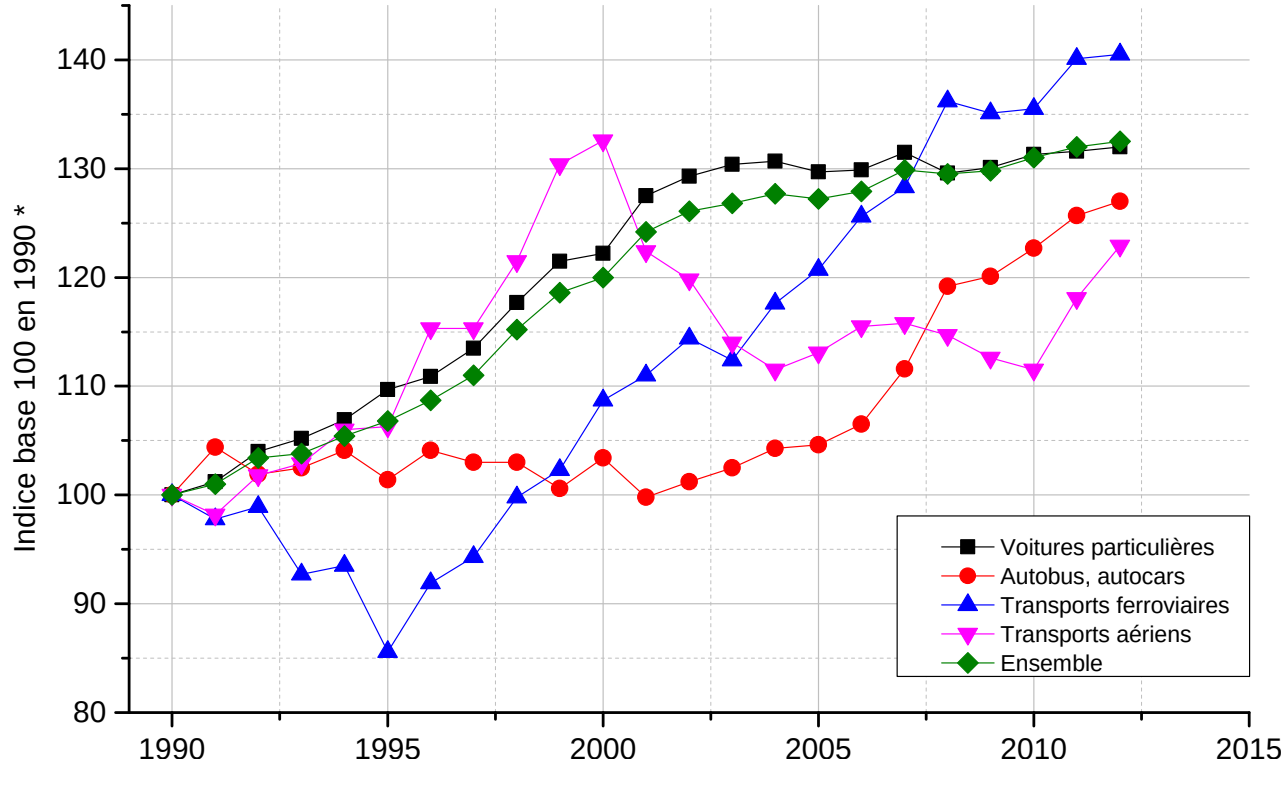



FIGURE I.3: Évolution des volumes de transports intérieurs de voyageurs de 1990 à 2012. Source : (INSEE, SOeS)

Du point de vue de l'offre, ces nouvelles pratiques de transit conduisent à des usages combinés de la voiture et des transports en commun, transforment les flux, influencent le paysage urbain et l'organisation de la vie quotidienne et encouragent l'utilisation des transport collectifs surtout dans les zones urbaines (Fig. I.4). De plus, l'amélioration de l'offre du transport renforce la dispersion géographique des déplacements, ce qui affecte les pratiques quotidiennes.

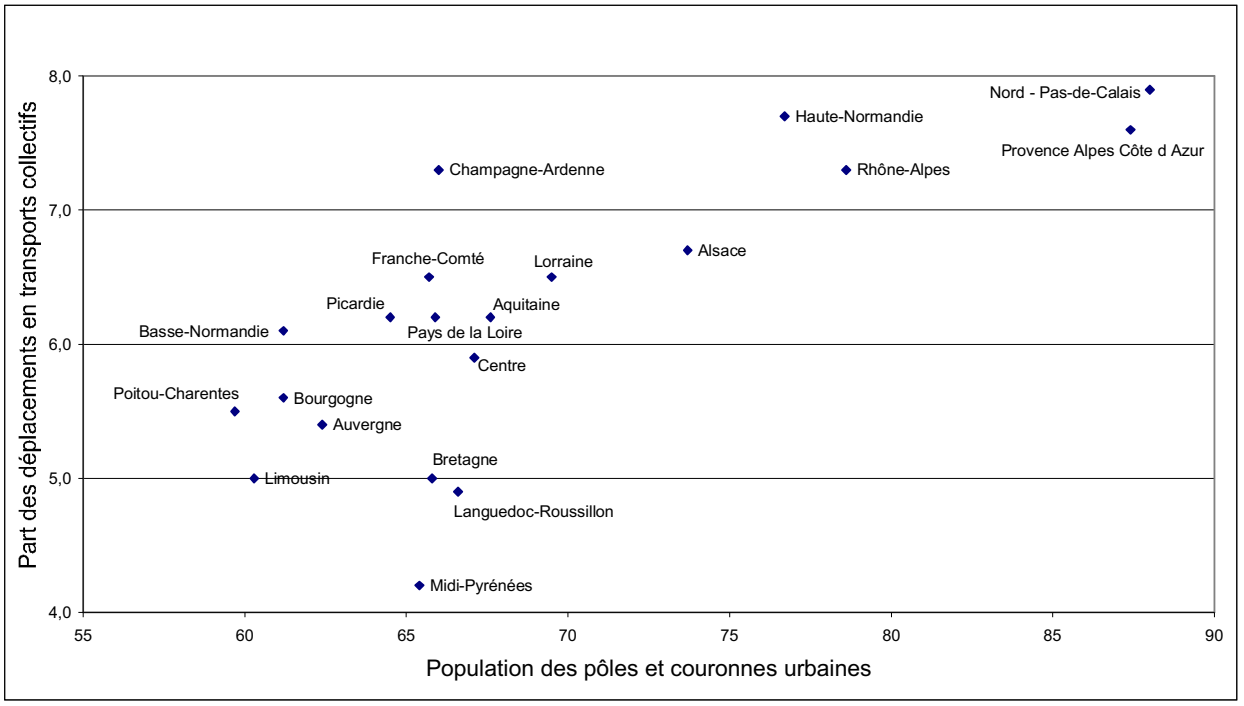



FIGURE I.4: Le recours aux transports en commun augmente avec l'urbanisation (en %) en 2008 (dernier rapport publié en 2010) [Armoogum et al., 2010]

### I.2.2 De la Mobilité Classique à la Mobilité Avancée

L'espace urbain est composé d'un ensemble de réseaux complexes et de systèmes qui interagissent et s'influencent mutuellement. Une planification urbaine inappropriée (ex. mettre différents modes de transport en concurrence) peut causer plusieurs problèmes dans le système de transport global : distribution inéquitable des coûts et des profits ; difficultés financière pour les ménages, gouvernements et entreprises ; inefficacité croissante à cause de la congestion de la circulation ; pollution, etc. Ces problèmes sont les conséquences d'une planification urbaine inappropriée vis-à-vis des avancées dans le domaine du transport.

Ces mauvaises conditions ont causé l'apparition de nouvelles politiques de planification de la mobilité urbaine durable. En effet, la planification du transport en Europe a progressé d'une manière continue en passant par plusieurs phases : de l'intermodalité qui consiste à combiner successivement plusieurs

modes de transport pour se déplacer, vers la multimodalité où différents modes de transport sont en concurrence et permettent un déplacement plus flexible, à la comodalité qui vise la complémentarité entre les différents modes de transport (publics et privés), et ce dans le but d'atteindre l'utilisation optimale et durable des ressources. La nouvelle forme d'évolution du transport n'est d'autre que l'intégration des services d'aide à la mobilité dans l'offre de déplacement afin d'assurer une meilleure QoS en terme de satisfaction des voyageurs et une meilleure gestion des ressources. Les tâches de recherche, de collecte et de distribution des ces services sont effectués par le moyen d'un système d'information qui génère du côté utilisateurs ce que nous appelons Information de Mobilité Avancée (IMA) introduite dans ce qui suit.

### I.2.3 Information de Mobilité Avancée

L'Information de Mobilité Avancée (IMA) issue d'un Système d'Information d'Aide à la Mobilité (SIAM) permet d'accompagner les utilisateurs des transports publics ou privés tout au long de leurs voyages (ex. réservations, horaires, recommandations, etc.). En outre, l'IMA permet d'enrichir l'expérience de voyage en rajoutant la possibilité d'avoir des informations sur des services connexes au domaine du transport (ex. tourisme, culture, environnement, etc.). L'IMA participe au rapprochement du client de son environnement quotidien et permet de mieux orienter ses choix dans l'ensemble de l'offre de mobilité. Par conséquence, la qualité de l'IMA en termes de disponibilité et de pertinence est de grande importance de nos jours et représente un critère principal pour l'évaluation de la qualité de service des fournisseurs de services de transport.

De ce fait, les acteurs des technologies sont focalisés sur les dimensions mobiles du quotidien et les solutions à apporter sous plusieurs formes : applications, services, plateformes de communication, etc. Ces initiatives améliorent la fluidité des déplacements par des outils d'information, de localisation et de guidage débouchant sur des produits comme la billettique, l'accessibilité dans les lieux publics, etc.

Néanmoins, la croissance exponentielle des services disponibles sur les Réseaux Distribués de Transport Comodal (RDTC) représente un véritable verrou technologique qui doit être levé afin de répondre d'une façon fiable et efficace aux besoins des clients du transport qui exigent des informations pertinentes, instantanées, interactives, et de meilleurs coûts possibles avant et pendant leurs déplacements.

Le client du service de transport a besoin de plusieurs informations en termes de planification, orientation, gestion de temps et suivi, lui permettant de :

– préparer son voyage à l'avance tout en comparant les différentes possibilités ;
– se localiser et s'orienter dans le réseau de transport et tout au long de son déplacement ;

– s'organiser et gagner en temps et en coût de déplacement ;
– anticiper les prochaines étapes de voyage et gérer en temps réel les perturbations qui peuvent se produire dans le réseau de transport.

Par conséquent, afin de répondre aux besoins du client, le système d'information d'aide à la mobilité doit être capable de fournir :

– un aide à la planification et à la prise de décision ;
– des informations de cartographie et de localisation ;
– des informations connexes en temps réel (météo, info-trafic, jeux, évènements, etc.) ;
– des informations sur l'état du réseau, sur les moyens d'anticiper les perturbations et sur les prochaines étapes d'un voyage.

### I.2.4 Classification des Informations de Mobilité Avancée

Quand le nombre, les caractéristiques et les fonctionnalités des services d'aide à la mobilité deviennent élevés, il est préférable d'avoir une classification claire de ces services afin de simplifier leur modélisation et leur intégration dans un système d'information d'aide à la mobilité avancée. En effet, les IMA peuvent être classifiées selon quatre différents aspects :

– *Aspect sémantique* : les IMA dans ce cas correspondent à différents domaines sémantiques. Ce type de classification se base essentiellement sur la catégorisation sémantique du contenu de l'information (ex. info-trafic, culture, actualités économiques et politiques, évènements sportifs ou culturels, etc.). En utilisant des approches de recherche sémantique, l'information peut être localisée plus efficacement [Paliwal et al., 2012, Ruta et al., 2014] et sélectionnée sémantiquement pour mieux correspondre aux besoins des utilisateurs [Kang et Sim, 2011] ;

– *Aspect temporel* : dans ce type de classification, les catégories sont générées à partir de la validité de l'information par rapport au temps. En effet l'information peut être dynamique, donc elle aura besoin d'être mise à jour continuellement (ex. info-trafic). Elle peut être statique (ex. localisation d'un monument historique) ou aussi semi-statique telle qu'une information sur un évènement culturel qui n'est valide que pendant une période bien déterminée ;

– *Aspect spatial* : dans ce type de classification, la position géographique de l'utilisateur est nécessaire pour définir les catégories de l'information. En effet, l'information peut dépendre de la zone géographique du client ; certaines informations sont plus pertinentes lorsqu'un client se trouve dans une zone $A$ et le seront moins lorsqu'il est dans une zone $B$ comme l'indique

[Bareth et al., 2010, Uzun et al., 2013, Groß et al., 2013]. A titre d'exemple, les utilisateurs dans un centre commercial s'intéressent plus à des informations sur des promotions que des informations météorologiques. Ces dernières peuvent être plus pratiques dans les gares ou les aéroports ;

– *Aspect d'intégration (granularité)* : cette notion est introduite par le domaine de recherche des architectures orientées service SOA [5]. En effet, les informations nécessaires pour les applications basées sur le web peuvent être élémentaires (dites aussi "atomiques") ou composées. Ceci veut dire que l'information correspondant à un but fonctionnel est le résultat de l'invocation d'un service unique ou d'une composition d'un ensemble de services. Un exemple d'information élémentaire est la météo ; un exemple d'information composée est le plan de voyage composé des réservations des transports, des hôtels, des musées, etc. A titre d'exemple, une plateforme d'automatisation du processus de composition des services est proposée dans [Paik et al., 2014]. Cette plateforme propose offre une orchestration automatisée de la composition dynamique des web services.

### I.2.5 Système d'Information d'Aide à la Mobilité

Un Système d'Information (SI) est défini par l'ensemble des éléments pour la gestion, le stockage, le traitement, le transport et la diffusion de l'information dans une organisation. Il vise à répondre à la chaîne de l'information qui se décompose en cinq objectifs : le recueil des données ; le traitement de ces données ; l'élaboration de l'information ; sa diffusion et son usage [Zgaya, 2007] [Land, 2004]. Par conséquent, un SIAM regroupe ces fonctionnalités et intègre des fonctions supplémentaires tout en étant robuste face à un nombre important d'utilisateurs simultanés. Ces fonctions permettent d'assister les clients de transport à consulter l'IMA dont ils ont besoin afin de les aider à décider de leurs plans de déplacement et de rendre ces déplacements plus agréables.

Comme expliqué précédemment (§I.2.2), le nombre des utilisateurs des moyens de transport publics (et/ou privés) augmente chaque année. Les politiques adoptées pour faire face à cette évolution continue ont subi des changements stratégiques majeurs. De nos jours, la mobilité des utilisateurs du transport collectif peut être améliorée grâce à l'introduction des technologies d'information et de communication. Nous parlons souvent de mobilité intelligente qui permet au client d'effectuer des déplacements plus organisés, plus agréables, plus rentables et plus écologiques. L'informatique ubiquitaire (§I.4) ainsi que la technologie mobile représentent les technologies clés de cette mobilité avancée. Le rôle de ces technologies est de permettre à un SIAM de fournir les IMA d'une manière efficace, flexible et continue.

5. Service Oriented Architecture

Le travail proposé dans cette thèse se positionne dans cette optique de modélisation et d'optimisation d'un SIAM. L'originalité de ce travail réside dans le fait que les SI conventionnels, présentés dans la partie suivante, proposent des services et des solutions techniques limités en comparaison avec le système proposé dans cette thèse qui se situe dans un contexte d'espace ubiquitaire. Dans la partie suivante, nous présentons quelques exemples de systèmes d'information conventionnels dans le domaine du transport aux échelles régionale et nationale.

## I.3 Systèmes d'Information Conventionnels dans le Domaine du Transport

Les systèmes d'information d'aide au déplacement existants offrent généralement des solutions de déplacement monomodales ou multimodales. Ils sont souvent présentés sous forme de sites web ou portails qui centralisent les informations et qui peuvent présenter des difficultés d'utilisation et quelques ambiguïtés pour de nouveaux visiteurs. Certains portails sont statiques, nous présentons quelques uns dans la suite, or le client a besoin d'une information temps-réel surtout dans les cas des grèves, incidents, suspensions de lignes, etc. Dans ce travail nous en proposons un système d'information capable d'optimiser la recherche, collecter, composer et distribuer les IMA d'une manière dynamique à un nombre important d'utilisateurs mobiles. Le système proposé est élastique dans le sens où il s'adapte bien à l'évolution dramatique des requêtes utilisateurs simultanées tout en optimisant les temps de réponse, les coûts et les quantités de données transférées des IMA.

### I.3.1 A l'Échelle Régionale

A l'échelle régionale, il existe plusieurs opérateurs de transport tels que :

- Transpole [6] de Lille Métropole Communauté urbaine ;
- La Régie Autonome des Transports Parisiens (RATP) [7] de l'Ile-de-France ;
- Transports en Commun Lyonnais (TCL) [8] de Lyon ;
- LePilote [9] de Marseille et des Bouches du Rhône ;
- etc.

Chaque opérateur de ces derniers couvre une zone limitée, et fonctionne indépendamment des autres opérateurs. Afin de s'adapter aux nouvelles tendances de déplacement où le voyageur parcourt des

6. http://www.transpole.fr/
7. http://www.ratp.fr/
8. http://www.tcl.fr/
9. http://www.lepilote.com/

distances plus longues (ce qui nécessite parfois l'agrégation de plusieurs opérateurs de transport au cours d'un même déplacement) de nouvelles politiques de gestion de transport à l'échelle nationale ont été menées dans le cadre de la PREDIM [10]. Lancée en 2001, la PREDIM regroupe plusieurs acteurs : État, autorités et exploitants de Transport, gestionnaires des infrastructures, industriels de ce secteur, opérateurs Télécoms, laboratoires de recherche, etc.

La plateforme PREDIM a été créée dans le cadre du PREDIT [11] dans le but de regrouper tous ces acteurs susceptibles d'enrichir l'information multimodale et de la rendre plus adéquate et souple aux besoins des utilisateurs. L'objectif du PREDIT est de fournir une information de qualité capable d'orienter les voyageurs vers le mode de transport le mieux adapté à leurs besoins (ex. offrir des possibilités de déplacement autres que le déplacement automobile individuel). Ainsi, l'enjeu de cette plateforme est la composition, l'organisation et la diffusion des services et des données de plusieurs réseaux de transport afin de garantir la complémentarité entre les différents modes de transport.

### I.3.2 A l'Échelle Nationale

**L'exemple Allemand : DELFI** [12]. Durchganigige Elektronische Fahrplan Information (DELFI) (Information horaire électronique continue) est un projet allemand initié en 1994 par le Ministère Fédéral des Transports. Le projet a été réalisé en plusieurs étapes avant d'être introduit au public en 2004 [13]. DELFI couvre le calcul d'itinéraire "porte à porte" [14] avant le voyage. Le tableau I.1 résume les aspects traités par DELFI.

Table I.1: Récapitulatif des aspects traités par DELFI

| **Modes de Transport** | **Types de Correspondances** | **Zones Géographiques** | **Information Traitée** |
|---|---|---|---|
| – Transports Collectifs Urbains (TCU)<br>– Transports Collectifs non Urbains (TCNU) (i.e. transport ferroviaire seulement)<br>– 2 Roues (2R)<br>– Marche à Pied (MP) | – TCU vers/de TCU<br>– TCNU vers/deTCNU<br>– 2R vers/de TCU ou TCNU<br>– MP vers/deTCU ou TCNU | – Agglomération<br>– Département<br>– Région<br>– National | – Horaires<br>– Recherche d'itinéraire<br>– Tarifs |

L'architecture du système DELFI se base sur un modèle réparti CORBA [15] composée d'un réseau de serveurs d'information. Les serveurs sont locaux ou régionaux et opérés à travers des API. Chaque serveur du système est responsable du calcul des itinéraires dans une région bien définie. L'élaboration d'une solution globale nécessite l'intervention d'une entité centrale responsable de la composition de

10. Plateforme de Recherche et d'Expérimentation pour le Développement de l'Innovation dans la Mobilité
11. Programme de Recherche d'Expérimentation et de l'Innovation dans les Transports Terrestres
12. http://www.delfi.de/
13. Rapport technique: http://www.delfi.de./.../Delfi5Doc_v1_0.pdf
14. Un trajet porte-à-porte est obtenu par une recherche d'adresse à adresse pour tout voyage d'un point A à un point B, par opposition au voyage point-à-point reliant uniquement les gares ferroviaires, les stations de bus ou de métro.
15. Common Object Request Broker Architecture

la solution finale. En effet, il existe un ensemble d'informations qui sont fournies à chaque serveur et mises à jour continuellement afin d'assurer la compatibilité et le bon déroulement de la collaboration des différents points de calcul.

**L'exemple Suisse : SBB CFF FFS** [16]. Ce système est multimodal mais ne couvre que les transports en commun terrestres, et exclut les transports individuels. La recherche des itinéraires peut être effectuée en porte à porte. Il s'agit d'un système de renseignement qui propose des horaires, et permet une recherche d'itinéraires sur sa couverture géographique (tout le pays). Il propose aussi des services de billettique. La Fig. I.5 présente un exemple de planification entre la ville de Chur et la ville de Genève en utilisant le portail SBB.

SBB CFF FFS

**Votre demande de correspondance**

| Détails | Gare/Arrêt | Heure | Durée | Change | Voyage avec |
|---|---|---|---|---|---|
| **Correspondances du Lu, 15.12.14** | | | | | |
| 1 | Chur<br>Genève | dép. 07:09<br>arr. 11:16 | 4:07 | 1 | IC |
| 2 | Chur<br>Genève | dép. 07:16<br>arr. 11:43 | 4:27 | 2 | RE, ICN |
| 3 | Chur<br>Genève | dép. 08:09<br>arr. 12:16 | 4:07 | 1 | IC |
| 4 | Chur<br>Genève | dép. 08:39<br>arr. 12:43 | 4:04 | 1 | IC, ICN |

**légende**
Restaurant
Minibar
R Réservation possible
FA Voiture-familles avec aire de jeux
FZ Espace familles sans aire de jeux
BZ Espace affaires en 1re classe: Réservation possible
RZ Espace silence en 1re classe

FIGURE I.5: Exemple de planification en utilisant le portail SBB

L'architecture système est entièrement centralisée (à l'opposé du système DELFI) et se base sur une base de données centrale : la base de données du CFF (Chemins de Fer Fédéraux) qui présente le système national du déplacement ainsi que les bases de données des dix villes participantes (Genève, Lausanne, Zurich, Saint Gall, Bâle, Lucerne, Neuchâtel, Lugano, Bienne, Berne) qui représentent environ 80% de la population suisse.

16. http://www.cff.ch/home.html

**L'exemple Anglais : Transport Direct** [17]. Transport Direct est un portail de planification des déplacements, lancé par le Ministère des Transports Anglais depuis 2004 et mis hors service en 2014 (le ministère a jugé que les utilisateurs anglais possèdent actuellement suffisamment de choix et d'alternatives qui sont parfois plus performantes). Le système couvre la totalité du pays et permet de comparer selon les moyens de transport adaptés les itinéraires, les durées et les prix des déplacements. Les voitures privées, les taxis ainsi que les modes doux (marche à pied, roller, etc.) et les 2 roues sont aussi pris en compte. Le système offre aussi des services de billettique, ainsi le voyageur peut réserver et payer dès l'étape de planification. Ce système est capable aussi d'informer les usagers en temps réel sur les conditions de voyage (modification, pannes, perturbations..) avant le départ. Plusieurs autres services pouvaient être offerts grâce à ce système. La recherche des données et la composition des itinéraires ont été assurées grâce à deux serveurs :

– NPTG [18] : répertorie les pôles d'échanges nationaux pour permettre d'assurer les correspondances entres différents opérateurs et régions et faire le lien entre chaque réseau régional avec le reste du réseau national [19] ;
– NaPTAN [20] : répertorie l'ensemble des stations et des arrêts (les noeuds du réseau) pour assurer tout accès aux réseaux locaux et régionaux [21].

Transport Direct se basait sur une architecture répartie. En effet, des calculateurs locaux sont responsables de calculer les itinéraires chacun sur sa propre couverture géographique vers un pôle d'échange qui le relie à un deuxième calculateur d'un autre opérateur distant. Ainsi tous les calculateurs ont une vision globale du réseau national pour répondre aux requêtes des utilisateurs. Le système propose seulement des services de calculs d'itinéraires et nécessite l'intervention de l'utilisateur pour la sélection de la meilleure solution.

**L'exemple Néerlandais : 9292** [22]. 9292 est le système national hollandais pour l'information sur le transport multimodal. Ce système fournit les horaires pour des itinéraires de type porte à porte en combinant tous les modes de transport collectif.

Ce système offre aussi des services de billettique ainsi que différents services connexes. Son architecture centralise des données de plusieurs opérateurs de transport public. En plus de l'architecture centralisée du système, le choix d'une solution optimisée nécessite typiquement l'intervention de l'utilisateur. Le nom du portail est inspiré de l'ancien numéro 0900 9292 utilisé par un centre d'appel offrant des

17. Le portail est fermé le 30 septembre 2014
18. National Public Transport Gazetteer
19. Rapport technique: https://www.gov.uk/government/.../background-nptg.pdf
20. National Public Transport Access Node
21. Rapport technique: https://www.gov.uk/government/.../background-information-naptan.pdf
22. http://9292.nl/en

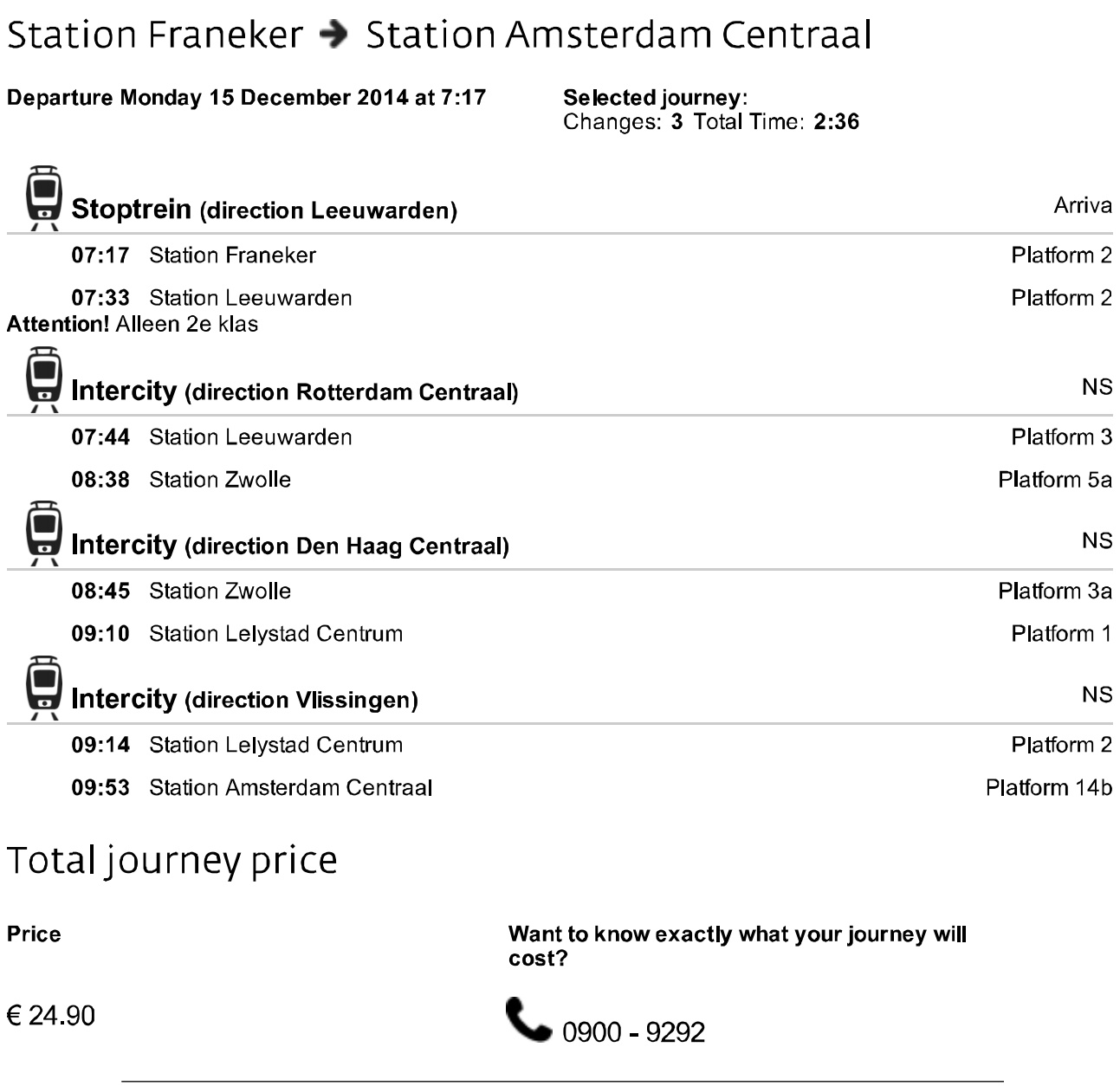



FIGURE I.6: Exemple de planification en utilisant le portail 9292

services de planification et d'aide au déplacement. La Fig. I.6 présente une capture d'écran du portail 9292.

### I.3.3 Discussion

Le secteur du transport est un secteur stratégique qui représente l'un des facteurs clés de succès pour une politique de développement d'un pays. Plus le pays est développé, plus la quantité d'information

multimodale s'accroit, ce qui la rend potentiellement mal gérée ou insuffisante aux yeux des utilisateurs. Les offres des systèmes d'aide aux déplacements actuels ne répondent pas exactement à cette panoplie de services qui ne cessent de se multiplier.

Ces systèmes restent limités dans le calcul des itinéraires et dans quelques services de billettique. Ces systèmes sont basés sur des systèmes de calcul centralisés ou parfois distribués (comme dans le cas du système DELFI) mais nécessites une base de données centralisée qui gère le système global. Toutefois on ne peut guère nier qu'ils présentent un moyen efficace d'orientation et d'aide à la décision. La nouvelle voie d'optimisation et d'amélioration des services de transport est certainement celle basée sur les nouvelles technologies d'information et de communication qui peuvent soutenir le voyageur avant et durant son trajet dans un contexte d'espace ubiquitaire communicant. Une architecture distribuée est capable de réduire la dépendance du système à des nœuds centraux pour le traitement des requêtes utilisateurs. De plus, ces dernières sont susceptibles d'augmenter exponentiellement pendant les périodes de pointe ce qui exige d'avoir un système adapté et capable d'asservir ces pics de demandes.

Une famille de systèmes d'information étendus et basés sur des architectures distribués sera présentée dans le prochain chapitre. Dans la suite, nous présentons le paradigme *informatique ubiquitaire*. Nous examinons en particulier les applications des TIC dans le domaine de l'aide à la mobilité avancée.

## I.4 Aide à la Mobilité Avancée dans les Espaces Ubiquitaires

Selon l'AFSCM [23] (Association Française du Sans Contact Mobile) et le Forum SMSC [24] (Services Mobiles Sans Contact) le parcours touristique NFC [25] à Nice, mis en service en 2010, permet l'accès à une base de données regroupant un ensemble d'informations historiques, culturelles ou touristiques dans 16 monuments du Vieux-Nice à travers des terminaux NFC. Ce projet fait partie de l'initiative intitulée Citizy [26] qui regroupe plusieurs acteurs : banques, opérateurs de téléphonie, services de transport et industriels. Au cours de cette initiative, 4000 téléphones mobiles dotés de lecteur/capteur NFC et compatibles avec les normes du projet ont été distribués pour permettre aux visiteurs et résidents de Nice de payer leurs transports et récupérer les horaires des bus et des trains [Pesonen et Horster, 2012].

La vague mondiale d'adoption du sans contact dans différents domaines (transport, santé, tourisme, etc.) fait partie du concept de l'informatique ubiquitaire et de l'intelligence ambiante. En effet, les informations contextuelles (§I.4.3.1) de l'environnement physique sont de plus en plus utilisées pour

23. http://www.afscm.org/
24. http://www.forum-smsc.org/
25. Near Field Communication (communication en champ proche)
26. http://www.cityzi.fr/

bâtir des applications de plus en plus "sensibles". Nous qualifions par le terme "sensible" toute entité capable de faire usage des "informations contextuelles" de son environnement afin d'optimiser et adapter son comportement. Ces informations sont collectées de différentes manières et en utilisant différents dispositifs. Le plus important est le fait que ces données soient partagées pour garantir une compréhension globale de l'environnement. Dans cette partie du chapitre, nous examinons le concept de l'informatique ubiquitaire et exposons quelques travaux de recherche réalisés dans ce domaine.

### I.4.1 Concepts et Définitions

#### I.4.1.1 Paradigme Informatique Ubiquitaire

Le paradigme informatique ubiquitaire ou "ubicomp", est un nouveau concept qui commence à prendre de l'ampleur sur les plans académique et industriel ces dernières années. En effet, notre environnement quotidien (ex. espace de travail, moyen de transport, domicile, etc.) intègre de plus en plus d'entités intelligentes capables de communiquer et de prendre des décisions. Une forme d'intelligence peut être créée à partir des données contextuelles (température, volume, poids, luminosité, humidité, distance, altitude, etc.) collectées de l'environnement. Ces données peuvent être par la suite utilisées par différentes applications que nous appelons "sensibles" [Salma, 2014, Yassine, 2011]. D'une façon plus générale, on qualifie par le terme "sensible" toute entité capable de faire usage des informations contextuelles de son environnement afin d'optimiser et/ou d'adapter son comportement. Ces informations sont collectées de différentes manières en utilisant différents dispositifs. Le plus important est le fait que ces données soient partagées pour garantir une compréhension globale de l'environnement.

La vision globale nécessite la coopération des nouvelles TIC en dépit de leur hétérogénéité ce qui nous oblige à prendre en compte l'aspect d'*ouverture* et à préparer les mécanismes de *collaboration* et d'*interopérabilité* lors de la conception de nos futurs SI.

#### I.4.1.2 Espace Ubiquitaire

Un espace ubiquitaire est un espace qui comporte des ordinateurs et des capteurs intégrés naturellement dans des objets de cet espace pour traiter et fournir des informations, des services et des ressources en temps réel n'importe quand et n'importe où. Selon Mark Weiser [Weiser, 1993], scientifique en chef à Xerox PARC, l'Informatique Ubiquitaire et la réalité virtuelle sont deux paradigmes opposés. En effet, la réalité virtuelle nous amène au sein d'un monde virtuel créé par l'ordinateur. Par contre l'Ubicomp amène les ordinateurs à notre monde réel et l'enrichit avec des informations.

Ce concept possède plusieurs dénominations : l'informatique omniprésente, l'informatique invisible, l'informatique sensible, l'informatique calme, l'intelligence ambiante [Stajano, 2010]. Il est utile de préciser que l'intelligence ambiante peut être considérée comme un domaine à part qui s'appuie sur les avantages de l'informatique ubiquitaire pour développer des systèmes plus sensibles et capables de s'adapter automatiquement aux utilisateurs et à leurs préférences en proposant des services plus spécialisés et des applications plus intuitives. On parle par exemple de maisons intelligentes, salles de conférences intelligentes, etc.

### I.4.2 Motivations, Outils et Limitations

L'Ubicomp fournit des applications capables d'assurer un support intuitif et transparent aux utilisateurs. Ces applications combinent les fonctionnalités d'un nombre de dispositifs dont la plupart sont intégrés naturellement dans l'espace. Ces dispositifs sont équipés de différents capteurs qui leur permettent de percevoir l'état de leur environnement. Ainsi, par le moyen de la communication sans fil, ils peuvent partager leurs perceptions et les combiner pour mieux comprendre et agir. Cette conscience sera traduite par des modèles de représentation de l'environnement qui permettent de développer une intelligence collective capable de comprendre les évènements présents et passés pour pouvoir prédire les états futurs de cet environnement. Ces applications doivent préserver le confort des utilisateurs en s'adaptant à leurs préférences. Dans ce contexte, les applications ubiquitaires sont très attractives du point de vue client.

De nos jours, l'Ubicomp se manifeste dans le phénomène de démocratisation universelle des technologies mobiles (Smartphones, PDA, ordinateurs portables, systèmes RFID et NFC, QR-Code, caméras, consoles de jeux, etc.) en créant un écosystème de communication mobile où on peut retrouver des supports applicatifs dotés d'une certaine sensibilité vis-à-vis de l'environnement et issus de l'intelligence artificielle.

Ce mouvement vers la technologie mobile et la substitution des ordinateurs fixes par des Terminaux à Ressources Limitées (TRL) engendre d'énormes verrous technologiques tels que : l'énergie limitée, l'hétérogénéité des dispositifs et des formats des données, le grand nombre de données et des utilisateurs, la connectivité intermittente, les incertitudes des capteurs, les changements continus des besoins et des préférences des utilisateurs, les interfaces limitées, etc.

Dans la suite, nous présentons un l'état de l'art sur les TIC dans le contexte de la mobilité avancée. Nous présentons en particulier la définition de l'information contextuelle nécessaire pour adapter les applications à leur environnement, le concept des réseaux de capteurs responsables de collecter

ces informations ainsi que les réseaux MANET et VANET pour la communication entre des nœuds mobiles.

### I.4.3 Technologies d'Information et de Communication au Profit de la Mobilité Avancée

Ce paragraphe présente deux exemples d'application des TIC. Le premier exemple illustre les systèmes de réseaux de capteurs sans fil qui permettent la détection distribuée, le calcul et la communication sans-fil afin de construire des applications dynamiques et sensibles aux changements qui se produisent dans l'environnement de l'utilisateur. Le deuxième exemple illustre les architectures réseaux MANET et VANET qui permettent d'établir un support de communication pour un ensemble de nœuds mobiles afin de développer l'aspect collaboratif des applications dans un environnement ubiquitaire.

#### I.4.3.1 Information Contextuelle et Réseaux de Capteurs sans fil

L'information contextuelle peut être définie comme étant la description de l'état, les applications, l'environnement, et la situation d'une entité donnée dans un système. Selon [Debes et al., 2005], il existe trois classes de contextes :

1. le contexte de l'utilisateur (ex. identité, position, date et heure, etc.) ;
2. le contexte du terminal (ex. identité, position, angle d'inclinaison, altitude, vitesse, etc.) ;
3. le contexte du réseau de communication (ex. débit, disponibilité, QoS, bande passante, etc.).

En effet, plusieurs informations peuvent être collectées de l'environnement des individus, des animaux ou des objets afin de créer des applications plus sensibles de leur environnement. Plusieurs exemples d'informations contextuelles peuvent être cités :

– *identité* : cette information a pour but de caractériser chaque entité par un identifiant qui doit être unique dans l'espace des noms de l'application ;
– *localisation* : cette information caractérise la position géographique, l'orientation ainsi que les distances avec les objets de l'environnement d'une entité donnée ;
– *état* : cette information décrit les propriétés des entités d'un environnement donné telles que la température, l'altitude, la vitesse, l'accélération, la puissance, etc.

Ces informations contextuelles sont collectées par le moyen de capteurs adaptés pour chaque classe et type d'information afin de permettre à l'environnement quotidien des utilisateurs d'interagir avec

eux et leurs fournir des services utiles et innovants. Les réseaux de capteurs sans-fil WSN sont l'un des premiers exemples concrets du concept de l'Ubicomp où des dispositifs de détection et de traitement de petites tailles, intelligents et à prix réduit vont pénétrer notre environnement quotidien en l'augmentant d'informations et d'intelligence.

Les WSN reposent sur la notion de l'interconnexion de plusieurs nœuds de calcul ou de détection dans le but d'agréger des données contextuelles afin d'avoir une vision globale d'un environnement donné et adapter les paramètres du système aux changements qui peuvent se produire dans cet environnement. Selon [Montoya et al., 2010], un réseau de capteurs est un système qui consiste en des milliers d'unités miniaturisées appelées nœuds de détection. La fonction principale de ces détecteurs est la surveillance, l'enregistrement et la notification des autres unités par une condition spécifique à différents endroits. La Fig. I.7 présente un exemple d'un WSN.

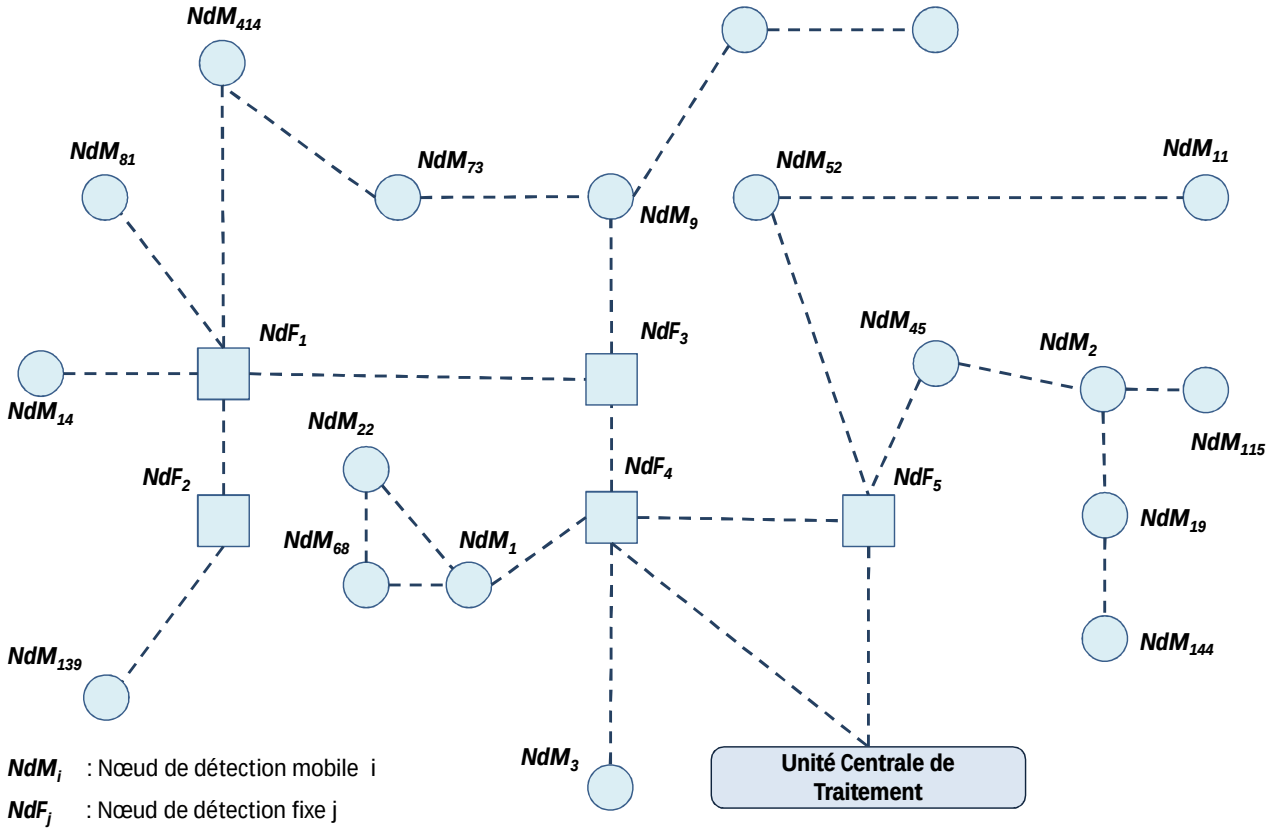



FIGURE I.7: Exemple d'un système de WSN

Les WSN combinent la détection distribuée, le calcul et la communication sans-fil. Ces réseaux permettent de relier des technologies tels qu'Internet à des applications distantes de divers domaines : les changements climatiques, la pollution, la surveillance militaire, la gestion et le contrôle des espaces de travail étendus, l'étude des comportements de certaines espèces dans la nature, etc. Les données

contextuelles récoltées par un WSN sont essentielles pour créer des applications sensibles, capables de s'adapter aux besoins des utilisateurs d'une manière dynamique et offrir des services en temps réel.

Plusieurs plateformes ont été mises en œuvre pour l'étude des WSN (IoT-LAB [27], WISEBED [28], INDRIYA [29], BigNet [30], etc.). Nous citons l'exemple de la plateforme IoT-LAB (anciennement appelée SensLAB). IoT-LAB est distribuée sur 6 sites en France (Inria Lille, Inria Grenoble, ICube Strasbourg, Inria Rennes, Rocquencourt, Institut Mines-Télécom Paris) avec une totalité de 2728 nœuds de détection sans fil. Elle offre une interface de programmation (API) qui permet aux développeurs de rédiger des scripts pour soumettre, relancer ou arrêter des expériences.

La supervision d'une application est totalement indépendante du programme déployé par l'utilisateur. En effet, la supervision externe offre un accès précis et en temps réel à des paramètres fondamentaux d'un WSN tels que la consommation énergétique et les activités radio. D'autre part, le développeur possède un contrôle temps réel de ses expérimentations. IoT-LAB offre un ensemble de commandes qui peuvent influencer l'environnement de l'application (ex. activer/éteindre des nœuds pour imiter des plantages, envoyer des faux signaux pour imiter le bruit radio, etc.).

Plusieurs applications ont été déployées sur la plateforme IoT-LAB. Nous citons à titre d'exemple l'application test des protocoles de géolocalisation des animaux dans la nature [Burin des Roziers et al., 2011]. Trois types de nœuds ont été utilisés : des "sink", des "anchor" (qui sont tous les deux fixes) et des nœuds mobiles. Les nœuds anchor enregistrent l'identifiant RSSI (Received Signal Strength Indicator) du nœud mobile ainsi que la date. Toutes les informations collectées par les anchor vont être acheminées vers le(s) sink. Ce dernier est connecté à un ordinateur central qui calcule les coordonnées du nœud mobile en se basant sur ces informations. Une carte-historique des comportements de migration des animaux est délivré à la fin de l'expérience.

Dans le paragraphe nous présentons les systèmes MANET et VANET qui permettent de créer une plateforme de collaboration entre plusieurs nœuds qui sont spécialement mobiles.

#### I.4.3.2 MANET et VANET

Les technologies de l'information et de la communication (TIC) sont introduites dans plusieurs applications dans différents domaines, en particulier, le transport et l'aide à la mobilité. En se basant sur des architectures de type pair-à-pair (chapitre II) l'architecture MANET (Mobile Ad hoc NETwork)

27. https://www.iot-lab.info/
28. http://www.wisebed.eu/
29. http://indriya.comp.nus.edu.sg/motelab/html/index.php
30. http://issnip.unimelb.edu.au/research_program/sensor_networks/infrastructure/bignet_testbed

et sa variante VANET (Vehicular Ad hoc NETwork) font partie de plusieurs projets de recherche qui vise à adapter les TIC aux contextes de l'intelligence ambiante et d'aide à la mobilité avancée. En effet, ces deux derniers reposent sur des architectures P2P où chaque nœud possède les deux comportements : client et serveur.

**MANET**. Les réseaux ad hoc mobiles font partie de la famille des réseaux P2P non structurés [Sun, 2001]. Un MANET est un ensemble de nœuds mobiles indépendants qui peuvent communiquer en utilisant des ondes radio (ex. le standard IEEE 802.15.1 (Bluetooth), le standard IEEE 802.11 (Wifi)) et sont capables de s'organiser d'une manière dynamique n'importe où et n'importe quand. Les nœuds mobiles qui sont mutuellement dans la zone de couverture de chacun peuvent communiquer directement, tandis que d'autres ont besoin de nœuds intermédiaires pour acheminer leurs requêtes. Chaque nœud dispose d'une interface sans fil qui lui permet de communiquer avec les autres nœuds. Ces réseaux sont entièrement distribués, et peuvent opérer sans l'aide d'une infrastructure fixe de points d'accès.

**VANET**. Les réseaux ad hoc véhiculaires sont considérés parmi les types les plus prometteurs de réseaux mobiles ad hoc par la communauté de recherche des réseaux sans fil. En effet, chaque véhicule est un nœud dans la topologie VANET. Ces nœuds sont équipés avec des DSRC [31] qui communiquent les uns avec les autres, chacun selon sa zone de couverture. Les systèmes VANET peuvent communiquer avec Internet ainsi qu'avec d'autres objets ou véhicules en utilisant des RSU [32]. En effet, les réseau VANET s'inscrivent dans la thématique des systèmes de transport intelligents et coopératifs. La Fig. I.8 présente une illustration d'un système VANET.

Dans [Tsukada, 2011], l'auteur traite le problème de la gestion de la communication entre des systèmes de transport intelligents afin de sélectionner le meilleur chemin de communication disponible. Dans [Liu et al., 2014], un système de communication temps réel entre les bus est afin d'échanger des informations permettant d'assister les conducteurs des moyens de transport public. Dans [Fathy et al., 2012], la QoS dans un système VANET est traitée pour optimiser le temps de réponse des communications entre un véhicule et une infrastructure fixe RSU appelé RBN [33]. Dans [Harrabi et al., 2013], Harrabi propose une modélisation orientée agent pour l'optimisation du processus de routages des trames de données et le choix du meilleur chemin dans un système VANET.

Enfin, les constructeurs de véhicules en Europe ont initié le CAR 2 CAR Communication Consortium (C2C-CC) [34] qui est une organisation à but non lucratif qui a pour objectif la standardisation des

31. Dedicated Short Range Communication Radio
32. Road Side Units (des unités sur le bord de la route)
33. Road Side Backbone Network
34. https://www.car-2-car.org

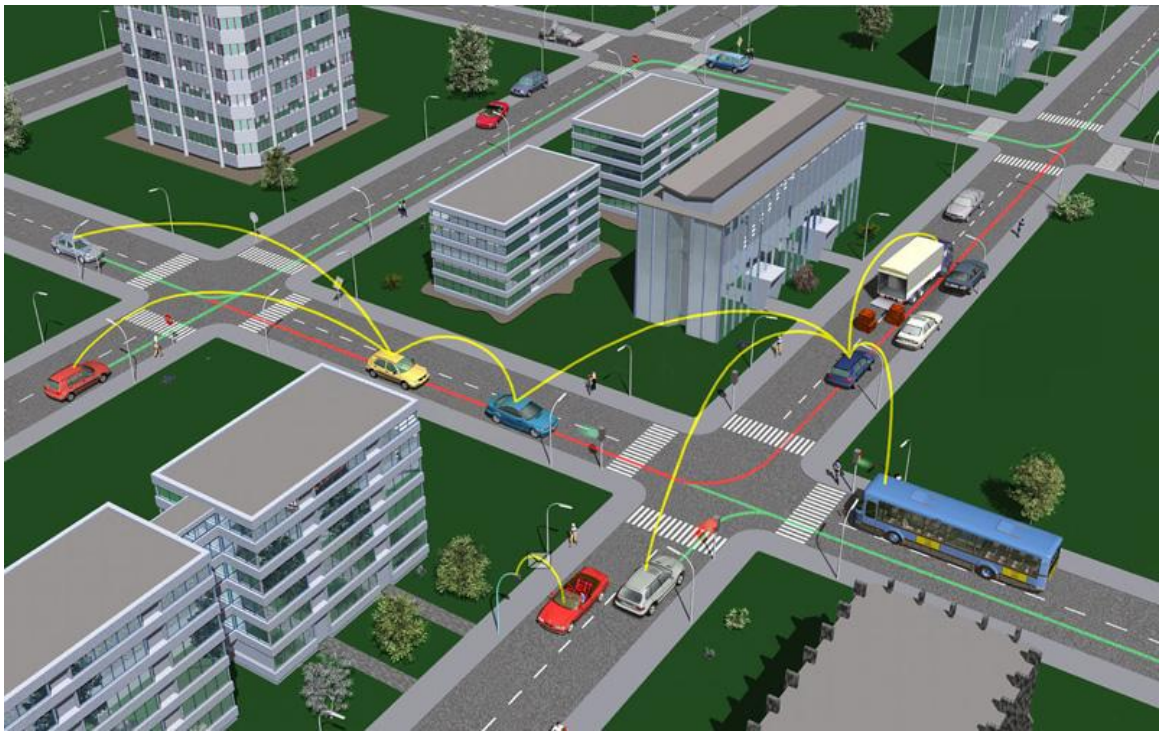


FIGURE I.8: Illustration d'un système VANET. Source : car-2-car.org

technologies de communication véhicule-à-véhicule et véhicule-à-infrastructure en Europe dans une action de recherche et développement stratégique dans le domaine de VANET.

#### I.4.3.3 Synthèse et Positionnement

L'introduction massive des technologies de l'information et de la communication (TIC) dans notre environnement quotidien a permit de créer de nouveaux concepts et applications à valeurs ajoutées élevées. Par exemple, les WSN permettent le suivi et la protection des animaux en voie de disparition dans la nature ; une fois appliqués, les réseaux VANET permettent une meilleure gestion de la congestion du trafic et la prévention des accidents routiers. En effet, ces technologies représentent le support de transport des informations d'aide à la mobilité dans un réseau distribué.

Les travaux de thèse présentés dans ce rapport se situent au niveau applicatif et proposent un système qui permet d'optimiser la recherche et la distribution des informations d'aide à la mobilité. Nous précisons que parmi les perspectives de ce travail de thèse est de pouvoir tester le système développé, en particulier le protocole de négociation proposé dans le chapitre IV, sur un WSN afin d'analyser les performances de ce système et valider les concepts théoriques développés en émulant une situation réelle d'un espace ubiquitaire.

### I.4.4 Applications et Projets en Informatique Ubiquitaire

Dans cette section, nous présentons quelques exemples de projets de recherche dans le domaine de l'informatique ubiquitaire. A la fin de cette partie, nous introduisons un projet de recherche initié au sein de notre équipe de recherche réalisé au tour du stade Pierre Mauroy de la métropole lilloise.

#### I.4.4.1 PERSONA (Perceptive spaces promoting independent aging)

Lancé en 2007, sous tutelle de l'FP6-IST [35], le projet européen PERSONA [Fides-Valero et al., 2008, Tazari, 2010] consiste en une plateforme distribuée d'assistance de l'autonomie des personnes âgées dans un environnement intelligent. La plateforme se base principalement sur un intergiciel qui porte le même nom. En effet, PERSONA est basé sur une architecture pair-à-pair (les systèmes pair-à-pair seront exposés dans le prochain chapitre) où chaque nœud possède son propre intergiciel qui lui permet de découvrir les autres nœuds afin de communiquer et coopérer avec ces derniers. Cet intergiciel offre quatre types de bus de communication basés sur les évènements : entrée, sortie, contexte et service. Tous les nœuds sont reliés à travers ces bus et peuvent exploiter les services et les informations contextuelles offerts par chacun d'entre eux. L'intergiciel PERSONA est composé de quatre couches :

- une couche physique (la plus basse) responsable de la connexion, la découverte des services et la communication entre les différents nœuds. Cette couche permet une communication basée sur des protocoles tels que UPnP [36], le standard IEEE 802.15.1 (Bluetooth) et OSGi [37] ;
- une couche service responsable de la gestion des bus et de la découverte des autres pairs présents dans l'environnement ;
- une couche spécifique PERSONA responsable de la sérialisation des messages provenant des différents bus avant de les transmettre. La sérialisation consiste à transformer un message en un ensemble de trames afin de faciliter son transfert ;
- une couche applicative responsable de l'interfaçage entre les pairs du système. L'interface permet l'accès aux services, capteurs et applications proposées par chaque pair. Cette couche offre aussi des services de publication et découverte des fournisseurs de services.

35. http://ec.europa.eu/research/fp6/index_en.cfm
36. http://www.upnp.org/
37. http://www.osgi.org/Main/HomePage

### I.4.4.2 MUSIC

Lancé en 2006, MUSIC [38] [Rouvoy et al., 2008, Khan, 2010, Desruelle et al., 2010] (Self-adapting applications for mobile users in ubiquitous computing environments) est un projet européen financé par la commission européenne [39] (EU IST 035166). MUSIC rassemble des partenaires académiques (Université d'Oslo, Université de Cassel, etc.) et industriels (RATP, SINTEF, Hewlett Packard, Telecom Italia, etc.) spécialistes dans le domaine de la technologie mobile et les domaines connexes. L'objectif de MUSIC est la réalisation d'une plateforme open source de développement des applications mobiles dynamiques, auto-adaptatives et reconfigurables. De plus, MUSIC fournit une méthodologie de conception et de mise en œuvre d'un SI distribué auto-adaptatif dans les espaces ubiquitaires.

L'intergiciel MUSIC repose sur une architecture distribuée regroupant un maître et un ensemble d'esclaves [Khan, 2010]. Le processus d'adaptation d'une application est déclenché par le changement de contexte du nœud maître. A titre d'exemple, le service de connexion internet d'un premier utilisateur $U_1$, le service d'affichage d'un deuxième utilisateur $U_2$ et finalement le service de géolocalisation d'un troisième utilisateur $U_3$ peuvent être combinés afin de créer une application de navigation dans un domaine d'adaptabilité d'aide à la mobilité. L'architecture de MUSIC (Fig. I.9) est composée de 4 modules principaux [Rouvoy et al., 2008, Khan, 2010] :

- Gestionnaire d'adaptation : ce module est responsable du lancement de la procédure de planification et du contrôle du processus de reconfiguration d'une application suite à un changement du contexte (ex. position, MT, lieu de travail, activité, etc.). Ce module propose un ensemble d'heuristiques pour déterminer la meilleure configuration d'une application. Le module calcule les nouveaux paramètres des applications et crée des plans d'adaptation qui seront exécutés à travers l'exécuteur des configurations.
- Noyau : ce module propose un entrepôt de plans d'adaptation qui peuvent être utilisés pour des configurations récurrentes. Il propose aussi une interface de gestion des services qui permet au gestionnaire d'adaptation de consulter l'ensemble des services disponibles.
- Gestionnaire de contexte : ce module est responsable de la détection des changements du contexte et de la mise à jour le gestionnaire d'adaptation. Ce module peut aussi fournir des informations par rapport à la qualité de service requise pour un contexte donnée.
- Gestionnaire de QoS : en se basant sur les métadonnées des configurations associées à chaque plan d'adaptation d'une application, ce module est capable d'offrir des informations sur la QoS de

38. http://ist-music.eu/
39. http://ec.europa.eu/index_fr.htm

chaque plan. Ces informations seront utilisées afin d'évaluer les plans proposés par le gestionnaire d'adaptation.


FIGURE I.9: Illustration simplifiée des modules composant l'architecture de MUSIC

La Fig. I.10 présente une capture d'écran de l'application "Travel Assistant" (Assistant de Voyage) développée pour la RATP [40] en se basant sur l'architecture MUSIC.

### I.4.4.3 UBILA

Lancé en 2003 par le MAIC (Ministère des Affaires Intérieures et des Communications du Japon), le projet UBILA [Ohashi, 2010] (nuage en latin) a l'objectif d'instaurer les technologies nécessaires pour les réseaux omniprésents et fournir le support applicatif et serviciel nécessaire aux utilisateurs. L'architecture proposée consiste en un ensemble de réseaux à différentes fonctions (contrôle du réseau, contrôle des services, collecte du contexte et diffusion) et une plateforme d'accès ouverte. La plateforme UBILA utilise les informations contextuelles des utilisateurs afin de leur offrir des fonctions de service et de contrôle optimisées. La plateforme regroupe plusieurs services et prototypes d'intelligence ambiante. Nous en citons quelques uns [Ohashi, 2010] :

– OSNAP (Open Sensor Network Access Protocol) : étant donné que le protocole TCP/IP [41] n'est pas supporté par les réseaux de capteurs, le protocole OSNAP a été proposé par le "Ubiquitous Networking Forum [42]" pour résoudre ce problème. Ce protocole est basé sur le protocole SNMP (Simple Network Management Protocol) destiné aux dispositifs connectés. L'inconvénient de ce protocole est la longueur limitée des messages de communication [Heo et al., 2010].

– DOLPHIN : un prototype d'un système de localisation intérieure réalisé par l'Université de Tokyo basé sur la détection distribuée par ultrason. Ce prototype répond au besoin de localiser les personnes à l'intérieur des bâtiments sachant que la localisation extérieure est réalisée à l'aide d'un

40. http://www.ratp.fr/
41. Transmission Control Protocol/Internet Protocol
42. http://ubiquitous-forum.jp/index2.html


FIGURE I.10: Illustration de l'application Travel Assistant. Source : "MUSIC consortium" sur Google Play [43]

système GPS.

– PAVENET : un prototype de réseau de senseurs développé aussi à l'Université de Tokyo. Son objectif est de supporter les applications temps réel et d'offrir des outils de programmation plus flexibles (PAVENET OS). Ce système de capteurs est utilisé pour mesurer l'accélération des vibrations sismiques.

– CASTANET[Kawahara et al., 2007] : c'est une architecture de services sensibles basée sur le modèle REST. Elle permet de collecter les données contextuelles délivrées par les réseaux de capteurs et de commander des actionneurs en utilisant le protocole HTTP. CASTANET considère les informations des capteurs, les ressources des web services, les actionneurs et les interfaces utilisateur comme étant des ressources et leur associe des URI uniques. Les clients peuvent effectuer quatre types d'opérations sur les URI : GET, PUT, POST et DELETE. GET permet de récupérer des informations concernant l'état de la ressource pointée par un URI, PUT permet de mettre à jour ces informations, POST permet de générer des ressources et de les envoyer vers un URI (ex. envoyer un message vers un écran) et DELETE permet de supprimer une ressource d'un serveur.

43. https://play.google.com/store?hl=fr

– Lifelog : un système d'information développé par KDDI [44] (opérateur de télécommunication japonais) qui fonctionne sur les téléphones mobiles. Il consiste en une base de données centrale regroupant des informations sur les prix, les désignations, les origines, etc. des produits dans les grandes surfaces. Les utilisateurs accèdent à ces informations grâce à des TM adaptés en utilisant les technologies GPS, reconnaissance du QR-Code, identification RFID, etc., afin de récupérer sur le TM d'une manière à la fois facile et rapide les propriétés des produits dans les grandes surfaces.

#### I.4.4.4 Stade Pierre Mauroy : Futur Espace Ubiquitaire

Suite à une coopération entre le LAGIS [45] (Laboratoire d'Automatique, Génie Informatique et Signal), actuellement CRIStAL [46], et l'entreprise ARISONA Consulting & Technology [47] dans le cadre d'un projet OSEO [48], une veille technologique a été menée afin de définir les technologies clés et proposer des démarches et des outils de mise en œuvre d'un système intelligent d'aide à la mobilité. Ce dernier est centré autour du stade/aréna Pierre Mauroy situé à la Métropole lilloise [Bousselmi et al., 2012, Bousselmi et Amara, 2011, Bertaux et al., 2013]. En effet, après une enquête et des études pour l'identification des services pertinents à offrir aux utilisateurs de cet espace, un ensemble non exhaustif de services d'aide à la mobilité a été identifié. Ces services sont répartis sur quatre thèmes :

– les services dédiés au stade/aréna qui peuvent être utilisés directement dans cet espace (ex. U-Player, U-Goal) ;
– les services touristiques qui permettent d'améliorer l'attractivité touristique de la ville (ex. U-Travel, U-Map) ;
– les services d'aide à la mobilité dans la ville permettant l'interaction avec les fournisseurs des services de transport public (ex. U-GPS+, U-Parcking) ;
– les services culturels dans la ville qui sont en complémentarité avec les services touristiques (ex. U-Book).

Quelques services proposés dans cette études sont présentés dans l'annexe A de ce mémoire. Trois applications Android U-Fun, U-GPS et U-Player ont été réalisées dans le cadre de ce projet par des élèves de l'Ecole Centrale de Lille [Bertaux et al., 2013].

44. http://www.kddi.com/english/
45. http://lagis.cnrs.fr/
46. http://cristal.univ-lille.fr/
47. http://www.arizona-ct.com/
48. http://www.bpifrance.fr/

#### I.4.4.5 Synthèse et Positionnement

Les recherches effectuées dans le domaine de l'informatique ubiquitaire se concentrent sur la conception de plateformes qui offrent des services sensibles en se basant sur des informations contextuelles collectées depuis des réseaux de capteurs intégrés dans l'environnement. Ces informations permettent d'ajuster et d'optimiser d'une manière dynamique le comportement des applications ubiquitaires.

D'abord, le projet PERSONA (§I.4.4.1) propose une couche de communication adaptée pour les espaces spécialisés (ex. maisons de retraite, maisons de soin, salles de conférences, bibliothèques, etc.). L'architecture PERSONA permet une modélisation modulaire des composants communicants avec une gestion des évènements (ex. alertes de sécurité, notifications, demandes d'aide) provenant des capteurs intégrés dans l'espace ubiquitaire.

Ensuite, le projet MUSIC (§I.4.4.2) s'inscrit dans la thématique de l'informatique mobile où la gestion des informations contextuelles d'un espace ubiquitaire plus étendu que celui considéré dans le cas du projet PERSONA devient plus difficile. MUSIC est une approche de conception et de développement des applications auto-adaptatives. L'intergiciel proposé dans ce projet repose sur une architecture composée de quatre modules nécessaires pour satisfaire l'objectif d'adaptabilité dans un contexte d'ubiquité lors de la conception des applications mobiles. Une application de démonstration, adaptée au réseau de transport de la RATP [49], est développée dans le cadre de ce projet ; cette application permet le calcul d'itinéraires et le choix automatique de la connexion réseau.

Finalement, le projet UBILA (§I.4.4.3) s'inscrit dans une suite de projets initiée par le MIC [50] japonais dans le cadre du projet national de la planification stratégique en termes de TIC. Dans ce projet, plusieurs plateformes, prototypes, protocoles et solutions technologiques ont été proposés pour répondre à des problématiques de communication, de localisation, d'optimisation de la QoS, etc. D'autres projets similaires sont menés dans d'autres pays tels que la Corée du Sud et la République de Singapour, l'Union Européenne et les États Unis [Friedewald et Raabe, 2011].

Les plateformes présentées dans ce paragraphe proposent toutes des services spécifiques à des utilisateurs spécifiques et adaptés chacun pour un espace spécifique. Dans le cas de PERSONA, la solution proposé repose sur une architecture distribué qui ne prend pas en compte l'hétérogénéité des services et des dispositifs existants dans l'environnement. Dans le cas de UBILA-Lifelog, les services proposés par la plateforme sont multiples mais en même temps fournis par un serveur central qui gère toutes les bases de données du système. Le framework UBILA permet de concevoir des services plus adaptés

49. http://www.ratp.fr/
50. Ministère des Affaires intérieures et des Communications

aux espaces ubiquitaires mais n'offre pas de garanties sur les performances ni de ces services ni de la plateforme qui les héberge.

Le système proposé dans cette thèse se base sur une architecture qui *distribue* le traitement et le stockage des données et *centralise* une petite quantité d'informations qui permet d'aider les utilisateurs à découvrir et localiser les services d'une manière flexible, efficace et adaptée à la technologie mobile. Les services pris en compte par la plateforme proposée ne sont pas limités par une région géographique, une thématique, une spécialité ou une fonctionnalité. Ces services sont principalement orientés mobilité mais sans l'exclusion des services connexes (ex. culture, tourisme, loisir, etc.). Notre système est capable de supporter un nombre important de requêtes utilisateurs tout en optimisant la recherche des services demandés à partir du RDTC. Les terminaux mobiles des utilisateurs ne sont pas considérés comme des simples demandeurs de services mais plutôt participent à l'optimisation de la charge de communication dans le réseau. Ceci représente l'aspect original de ce travail.

En effet, dans notre système tous les terminaux mobiles des utilisateurs hébergent des agents. Ces derniers sont appelés des agents utilisateurs. Grâce à une approche de changement de rôle, en plus de leurs rôles en tant que clients simples (demandeurs de services), ces agents sont capables d'acquérir des rôles de fournisseurs de services. De plus, en se basant sur des algorithmes d'optimisation génétique combinés avec un système d'agents, les requêtes simultanées envoyées par les utilisateurs sont traitées en plusieurs étapes afin de garantir les meilleurs réponses en termes de temps de réponse, coûts et quantités de données transférées. Le système proposé est présenté plus en détail dans les chapitres III et IV de ce rapport.

Par ailleurs, le travail proposé dans cette thèse se situe dans le contexte de l'informatique ubiquitaire qui représente la force motrice des Technologies de l'Information et de la Communication (TIC) de demain.

## I.5 Problématique Traitée

Les informations d'assistance des voyageurs dans leurs déplacements sont offertes par l'intermédiaire d'une couche servicielle émergeante qui fait le lien entre le client et son environnement ubiquitaire en permettant une interaction bidirectionnelle en temps réel. En effet, les avancées technologiques au niveau de la miniaturisation de l'électronique ont permis la création de dispositifs électroniques et informatiques à différentes capacités de calcul et modes d'affichage pour devenir des éléments indispensables au quotidien. Ces dispositifs sont capables de communiquer pour fournir des services

et augmenter notre environnement d'informations et de connaissances. Dans ce contexte trois défis sont à relever :

- la gestion de la quantité importante de services disponibles, chacun avec un coût, un temps de réponse et une quantité de données transférées différents ;
- la gestion du nombre élevé des utilisateurs simultanés qui ont besoin d'un système robuste, efficace et capable de gérer la montée en charge ;
- la proposition d'une recherche automatisée des services. Étant donné que les mêmes services peuvent être proposés par plusieurs fournisseurs à différents coûts, délais de réponse, etc. l'utilisateur a besoin dans ce cas d'être assisté lors de la recherche des meilleurs services ;
- la topologie du réseau informatique distribué où les fournisseurs de services ne sont pas accessibles à travers une entité centrale. Dans ce cas, la solution apportée doit prendre en compte le caractère distribué et la dynamique élevée de l'environnement ;
- l'utilisation croissante des terminaux mobiles (Smartphones, Tablettes, Netbook, Laptops, etc.) qui possèdent des ressources de calcul, de stockage et une autonomie limitées ;
- la proposition de résultats d'un manière intelligente en se basant sur des algorithmes recherche et d'optimisation ;

Les services que propose notre système permettent de soutenir les voyageurs dans leurs déplacements en leur proposant une nouvelle expérience de mobilité intelligente. Ces services sont au cœur de la mobilité dans le but d'assister les utilisateurs et rendre leurs déplacements plus agréables.

## I.6 Conclusion

Le contexte de la mobilité urbaine ainsi que les avancées technologiques en termes d'informatique ubiquitaire permettant de réaliser une mobilité augmentée d'intelligence ont été présentés dans ce premier chapitre. Les espaces ubiquitaires sont capables de fournir des centaines de supports communicants qui permettent d'offrir des services n'importe où et n'importe quand. Ce nombre très important de services dans des tels espaces doivent être efficacement gérer par le moyen d'un système d'information qui doit être en mesure d'optimiser la recherche, la collecte et la distribution des données aux utilisateurs de ces espaces.

Dans ce chapitre, nous avons montré le positionnement de notre travail par rapport à quelques travaux et projets de recherche. En effet, nous proposons dans cette thèse une Plateforme de Recherche et de composition des Services d'Aide à la Mobilité (PRoSAM). Ce système permet aux utilisateurs

d'accéder à un ensemble de services d'une manière efficace et flexible. Par ailleurs, nous avons présentés un ensemble de verrous scientifiques qui sont traités dans ce travail.

Dans le chapitre suivant, nous présentons une étude bibliographique des modèles de systèmes étendus les plus réputés par leur capacité à monter en charge et gérer la simultanéité de nombre élevé de requêtes utilisateurs. Nous présentons à la fin une comparaison entre les différents systèmes et nous présentons notre choix par rapport à la méthodologie adoptée dans ce travail. L'alliance entre les systèmes d'agents et l'optimisation pour la modélisation et résolution des problèmes distribués fait partie de l'originalité du travail que nous présentons.

# Chapitre II

# Modélisation et Optimisation des Systèmes d'Information au Profit de la Mobilité Avancée

**Sommaire**

## II.1 Motivations

Afin de faciliter la mobilité urbaine des personnes, il est nécessaire de les assister pour avoir des déplacements dans les meilleures conditions possibles depuis la préparation du déplacement jusqu'à l'arrivée au point de destination. Il faut donc fournir au voyageurs toutes les informations nécessaires en temps réel en leur proposant divers services, qui peuvent aussi bien correspondre à des services de transport (une portion de chemin, une zone géographique, le trafic routier, etc.) ou à des services connexes (événements culturels, prévisions météorologiques, jeux, etc.). L'information fournie doit être pertinente, rapide et interactive en temps réel avant et au cours du déplacement et ce via n'importe quel dispositif de connexion (Smartphone, Tablette, ordinateur portable, PDA, etc.).

L'objectif à atteindre représente un véritable défi par rapport à la croissance exponentielle des services disponibles sur les réseaux distribués à grande échelle. En effet, une information demandée correspond à un service, qui peut être proposé différemment, par plusieurs fournisseurs d'information, en concurrence, avec différents coûts, temps de réponse et taille de données. Sachant qu'un ensemble de services peut être sollicité simultanément par un nombre très important d'utilisateurs, des mesures de conception et d'optimisation spécifiques doivent être prises en considération pour faire face aux risques d'effondrement d'un tel système d'information.

L'état de l'art présenté dans ce chapitre représente le fondement théorique et pratique de ce travail qui propose un système d'information distribué à base d'agents optimisateurs capable de gérer, rechercher et fournir d'une façon fiable et efficace des services d'aide à la mobilité avancée dans un environnement ubiquitaire.

La suite de ce chapitre est répartie sur quatre axes : Le premier axe présente un état de l'art sur les systèmes d'information étendus en particulier les systèmes pair-à-pair, informatique en nuage ("Cloud Computing") et grille informatique ("Grid Computing"). Le deuxième axe est consacré pour la présentation des systèmes multi-agents comme étant une méthodologie de modélisation et un outils d'abstraction des systèmes intelligents distribués. Le troisième axe se concentre sur l'alliance agent-optimisation qui représente l'atout principal de ce travail. En fin, suite à l'état de l'art présenté, le quatrième axe illustre une comparaison entre les différents systèmes étudiés dans cet état de l'art ainsi qu'un positionnement par rapport à ces systèmes. Par ailleurs, ce dernier axe conclut par rapport aux avantages des systèmes multi-agents dans le cas de notre problématique.

## II.2 État de l'Art sur les Systèmes d'Information Étendus

### II.2.1 Introduction

Les enjeux des systèmes d'information consistent principalement en l'ouverture, l'interopérabilité et l'adaptabilité dans un environnement distribué, hétérogène et dynamique. Dans les SI classiques, un serveur central ou une grappe avec un nombre limité de serveurs est essentielle pour accomplir plusieurs tâches, à titre d'exemple :

- identification des clients ;
- gestion et stockage de données ;
- services de routage pour les fournisseurs ;
- recherche et découverte de services pour les clients ordinaires ;
- business-to-business ;
- etc.

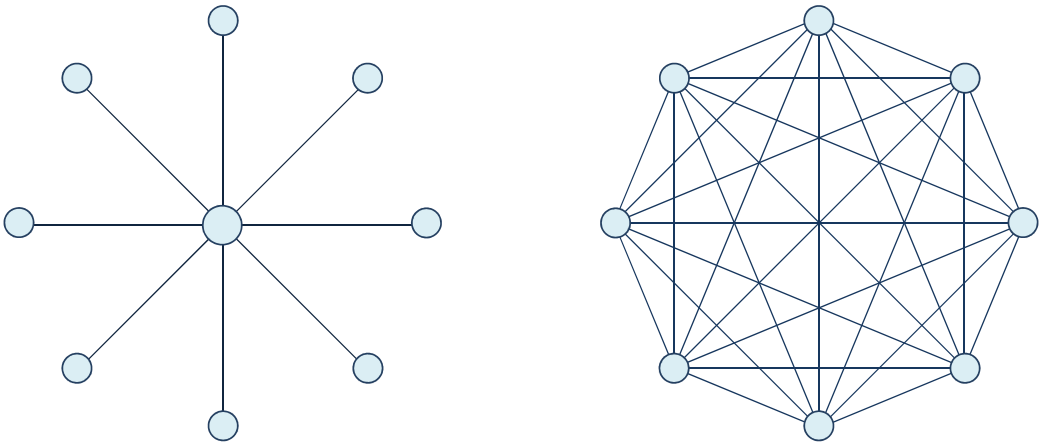
Par ailleurs, le nombre d'utilisateurs, la quantité de données ainsi que les flux d'information (clients-serveurs, serveurs-serveurs et clients-clients) en circulation sur le réseau augmentent d'une manière exponentielle.

### II.2.2 Classification des Systèmes d'Information Étendus

Dans un SI distribué, les opérations de stockage, traitement, diffusion, etc. peuvent être implémentées toutes ou en partie d'une manière distribuée. C'est à dire que la gestion de ces opérations n'est pas effectuée par une seule entité centrale mais plutôt distribuée sur plusieurs entités du système.

En effet, dans les systèmes dits "classiques" basés sur une architecture centralisée [Tamilarasi et Ramakrishnan, 2012, Liu et al., 2005], un serveur central est responsable des opérations d'authentification des clients, de recherche des services et de génération des réponses. Néanmoins, le nombre des utilisateurs de ces systèmes augmente continuellement. De plus, le contenu disponible sur le web en termes de services, applications, informations, données, etc. augmente d'une manière exponentielle. Ces services et ces informations peuvent être demandés à travers une large gamme hétérogène de dispositifs. Par conséquent, le concept de centralisation des informations atteint rapidement ses limites dans un tel contexte. L'adoption d'une architecture distribuée n'est pas un choix systématique pour s'adapter au contexte de l'ubiquité puisque ce type de système renferme un verrou scientifique majeur : la stratégie de recherche et de localisation des services.

En effet, il existe deux modèles d'architectures : centralisé et distribué. Ces derniers peuvent être combinés selon des compromis afin de s'adapter aux besoins de conception et optimiser les performances globales du système. La Fig. II.1 présente un exemple de deux architectures typiques : centralisée (basée sur un modèle Client/Serveur) et distribuée (basée sur un modèle Pair-à-Pair). Ces deux modèles d'architecture de SI ont contribué à l'invention de toutes une génération de SI à plusieurs échelles et domaines d'application.



(a) Architecture Centralisée (b) Architecture Distribuée

FIGURE II.1: Modèles d'architectures

L'état de l'art présenté dans cette partie nous permet de se positionner par rapport aux différents systèmes existants. Il nous permet aussi de définir la méthodologie adoptée pour résoudre la problématique traitée dans le cadre de cette thèse. Cet état de l'art présente trois types de systèmes :

- *les systèmes pair-à-pair (P2P)* : l'importance du concept P2P provient du fait qu'il représente la pierre angulaire de plusieurs systèmes distribués (ex. les grilles informatiques, les protocoles de routage et de découverte de services distribués, les réseaux de capteurs sans-fils, etc.). Il existe deux grandes familles de systèmes P2P : les P2P purs et les P2P hybrides (§II.2.3.1) ;
- *les grilles informatiques (GC)* : ce concept consiste à développer des applications distribuées afin d'agréger une énorme capacité de calcul et de stockage d'un ensemble de nœuds à faibles et moyennes performances (CPU, mémoire, bande passante, etc.). C'est à dire que le système de GC fait l'association d'un très grand nombre d'ordinateurs et d'unités de calcul et de stockage de moyennes et faibles puissances dans le but de créer un réseau interconnecté de nœuds de traitement comparable en puissance à celui d'un supercalculateur. Ce concept repose principalement sur les ressources des volontaires et des organismes scientifiques publics afin de contribuer à la résolution d'un problème complexe nécessitant des capacités de mémoire et des cycles de traitement très élevés (§II.2.3.2) ;

– *informatique dans les nuages (Cloud ou CC)* : ce concept a attiré l'attention des mondes académiques et industriels ces dernières années. L'idée consiste à fournir des espaces de travail scalables et à des coûts réduits aux entreprises et aux individus tout en garantissant la fiabilité, la continuité et la sécurité des services et des applications hébergés sur ces espaces. Le nuage en lui même peut être vu comme étant un service qui peut être fourni à trois échelles différentes : infrastructure, plateforme et application ; chacune correspond à des besoins très spécifiques de l'utilisateur (§II.2.3.3).

Les caractéristiques et les spécifications de ces trois types de systèmes étendus sont présentés en détail dans les paragraphes qui suivent. Ainsi, une étude comparative est présentée à la fin de ce chapitre afin de positionner les approches et les méthodes choisies pour développer la solution proposée dans le cadre de cette thèse par rapport à l'existant.

### II.2.3 Topologie des Systèmes d'Information Étendus

Dans ce paragraphe, nous présentons trois différentes topologies de systèmes étendus et nous introduisons par la suite les systèmes multi-agents. A la fin de ce chapitre nous présentons une étude comparative de ces différents systèmes.

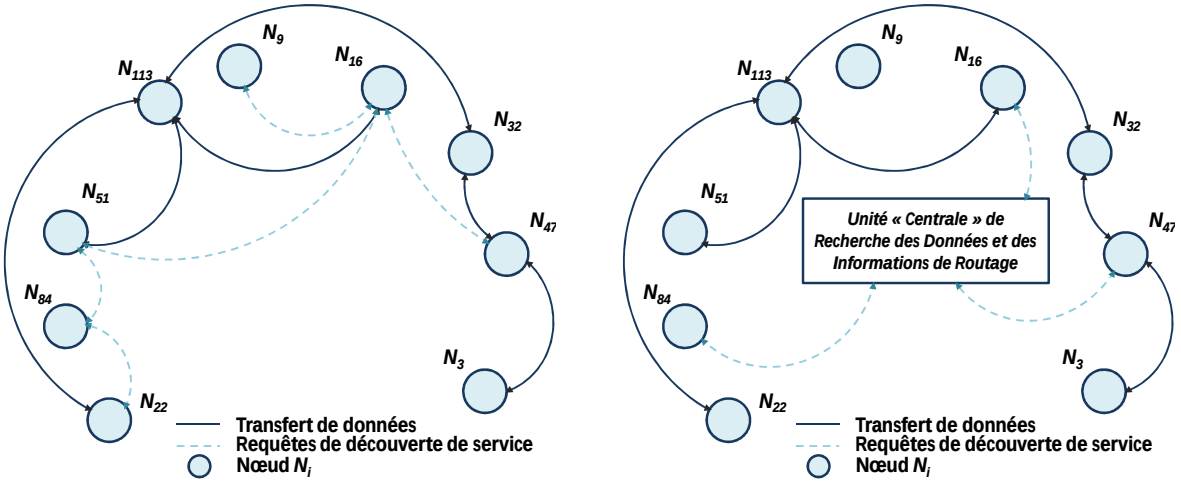
#### II.2.3.1 Systèmes Pair-à-Pair

Les systèmes P2P sont capables de supporter un nombre très élevé de sollicitations simultanées et peuvent fournir des services à des millions d'utilisateurs dans le monde entier [Pouwelse et al., 2005, Huang et al., 2008, Zhang et al., 2010, Zhang et al., 2009, Lua et al., 2005, Pourebrahimi et al., 2005, Merabti et El Rhalibi, 2004, Binzenhofer et al., 2006]. Ces architectures ont la capacité de s'organiser en dépit de la variation continue de la population des nœuds et des caractéristiques dynamiques du réseau (les nœuds se connectent et se déconnectent aléatoirement), la mobilité des utilisateurs, l'hétérogénéité des supports de communication, la variation de la charge, etc. En effet, les principales caractéristiques des architectures P2P sont la capacité de supporter la montée en charge ("scalability") ainsi que la distribution de l'accès croissant aux ressources [Lua et al., 2005].

Des opérations telles que l'administration et la maintenance n'appartiennent plus à une entité centrale mais sont plutôt distribuées entre les utilisateurs du système. Ces caractéristiques rendent les systèmes P2P convenables pour supporter les applications distribuées : bases de données, ressources de traitement et de stockage (RTS), aide à la décision, supervision, etc. Le verrou fondamental dans une architecture P2P est la localisation de l'information, appelée souvent la découverte des services.

Plusieurs protocoles ont été développés pour résoudre ce problème [Maymounkov et Mazières, 2002, Stoica et al., 2001, Han et al., 2014, Portmann et al., 2001].

Les systèmes P2P hybrides (Fig. II.2(b)) reposent principalement sur des registres centraux où s'inscrivent les différents fournisseurs de services. Ces derniers y publient leurs descriptions de service. Au contraire, les systèmes P2P purement décentralisés (Fig. II.2(a)) limitent l'influence des entités centrales et distribuent totalement les tâches de stockage et de recherche des données. Dans ce qui suit, nous présentons la définition d'un système P2P ainsi que ses fonctions principales.




(a) Système P2P Pur (b) Système P2P Hybride

FIGURE II.2: Exemple de systèmes P2P pur et hybride

### a. Définition d'un Système Pair-à-Pair

**Définition d'un Pair** : un pair peut être défini comme étant un élément d'un système de nœuds capable, à la fois, de lancer et de recevoir des message de recherche de services.

**Définition d'un Système Pair-à-Pair** : Il existe plusieurs définitions de systèmes P2P dans la littérature. Nous citons à titre d'exemple les définitions de [Roussopoulos et al., 2004] et [Schollmeier, 2001].

Selon [Roussopoulos et al., 2004], un système est qualifié de P2P s'il satisfait les trois critères suivants :

- Auto-organisé : les nœuds du système s'organisent d'une manière autonome par le moyen d'un processus de découverte des autres nœuds ou pairs du réseau ;
- Communication symétrique : les pairs du système sont considérés comme étant des égaux, ils peuvent à la fois offrir et demander des services en dépit des rôles de départ (client ou serveur) ;

– Décentralisé : les pairs du système n'ont besoin ni d'un nœud central de routage ni d'un nœud contrôle qui leur dicte leurs comportements individuels. Mais au contraire, les pairs sont autonomes et responsables de la découverte des autres pairs, des services ainsi que les ressources disponibles.

Par conséquent, un système distribué est qualifié de P2P s'il répond aux deux conditions suivantes :

– Les composants qui forment le système partagent leurs ressources afin de fournir les services pour lesquels le système est conçu. Par exemple, un réseau d'unités de traitement dont chacune offre une capacité de calcul donnée ;
– Les composants du système offrent et demandent à la fois des services les uns des autres. Par exemple, un réseau de stations météorologiques qui offre et demande des informations météorologiques afin d'établir des visions globales d'un environnement.

Afin de rejoindre un système P2P (i.e. devenir un pair du système ), le nœud doit établir une connexion avec un ou plusieurs pairs qui font déjà partie du système. Cette fonction s'appelle la fonction de découverte et permet aux nœuds de découvrir les pairs du système afin de communiquer avec eux.

**b. Fonctions d'un Système Pair-à-Pair**

Il faut faire la distinction entre les services et les fonctions d'un système P2P. En effet, les services sont d'une grande variété et dépendent de chaque application (ex. distribution de contenu et partage de fichiers, traitements intensifs et calcul distribué, etc.). Chaque système P2P possède deux fonctions principales qui gèrent la manière dont les pairs se connectent au système. Ces fonctions sont implémentées d'une manière centralisée dans le système :

– La fonction d'inscription : les nœuds qui rejoignent le système ont besoin d'obtenir des certificats valides pour pouvoir rejoindre le système. La fonction d'inscription gère l'authentification et l'autorisation des nœuds ;
– La fonction de découverte des pairs : afin de rejoindre un système P2P (i.e. devenir un pair du système), le nœud doit établir une connexion avec un ou plusieurs pairs qui font déjà partie du système. La fonction de découverte permet aux nœuds de découvrir les pairs du système afin de communiquer avec eux. Cette fonction est appelée *fonction de découverte de services* car le service proposé par un nœud est plus important que le nœud lui même.

Dans la littérature, plusieurs protocoles de découverte des services ont été proposés. Les fonctions de découverte des services utilisées dans les protocoles purement distribués et hybrides sont détaillées à travers des exemples proposés dans l'annexe B de ce mémoire.

#### II.2.3.2 Grilles Informatiques, "Grid Computing"

Les grilles informatiques (GC) [Giorgino et al., 2010, Anderson, 2004, Cappello et al., 2005, Camarasu-Pop et al., 2013] représentent une nouvelle famille de systèmes informatiques collaboratifs basés sur l'interconnexion de plusieurs unités de traitement et de stockage dans le but de surmonter les problèmes reliés aux architectures conventionnelles centralisées (coût très élevé de maintenance et de mise à jour des bases de données, sécurité, goulets d'étranglement, montée en charge, flexibilité et extensibilité, etc.). En utilisant Internet, certaines GC ont gagné de la popularité grâce à des outils de gestion très simples. En effet, il suffit uniquement de télécharger une application et de l'installer localement chez le client. Cette application a la capacité de détecter les périodes d'activité basse du processeur de la machine locale. Durant ce type de phases, l'application télécharge et traite un sous ensemble des données d'un projet. Une fois les traitements terminés, les résultats sont renvoyés au projet et le cycle reprend. Les GC de ce type se spécialisent dans une large gamme de projets qui peut aller de la recherche d'une vie extraterrestre à la recherche d'un traitement à des maladies comme le cancer ou le SIDA.

Les recherches dans le domaine du *Grid Computing* visent deux objectifs principaux :

- Fournir un accès à distance, sécurisé et omniprésent à des ressources de traitement et de stockage reliées aux TIC ;
- Rassembler et gérer les capacités de calcul et de stockage de plusieurs systèmes informatiques pour créer un système global performant qui rivalise avec les supercalculateurs.

**a. Définition d'une Grille Informatique**

Les grilles informatiques ou "Grid Computing" (GC) sont définies comme étant une infrastructure totalement distribuée [1], dynamiquement reconfigurable, évolutive et autonome, capable d'offrir un accès omniprésent, fiable, sécurisé et indépendant de la localisation des informations à un ensemble de services en encapsulant et virtualisant les ressources (traitement, stockage, données, etc.) dans le but de générer des connaissances [Laforenza, 2006]. Dans un système GC, les nœuds peuvent être gérés d'une manière centralisée où un serveur central choisit qui fait quoi, quand et pour combien de temps. Dans [Foster et al., 2001], Foster définit le problème des grilles comme étant un partage de ressource flexible, sécurisé et coordonné à travers une collection d'individus, d'organisations et de ressources où les challenges principaux sont l'unicité de l'authentification, l'autorisation, l'accès et la découverte des ressources. D'une manière générale, les grilles sont utilisées pour résoudre des problèmes scientifiques,

1. Dans le sens où, en comparaison avec un supercalculateur, le traitement est effectué par des centaines ou des milliers de nœuds distribués physiquement et/ou géographiquement. De plus, les nœuds d'un système GC n'interagissent pas forcément les uns avec les autres.

techniques ou entrepreneuriaux qui demandent une très grande capacité de traitement ou le traitement d'une très grande quantité de données.

**b. Exemples d'Application des Grille Informatique**

Le concept des GC a attiré beaucoup d'attention cette dernière décennie. En effet, des centaines de projets [2] sont élaborés autours de différents domaines (mathématiques, finance, cryptographie, bases de connaissance collaboratives, médecine, biosciences, etc.).

A l'échelle nationale, Grid*5000 [Cappello et al., 2005, Chakroun, 2013, Martin et al., 2014, Margery et al., 2014] est une plateforme de test et de validation des architectures de GC lancée en 2003 par le ministère de la recherche français.

A l'échelle internationale, BOINC [3] (Berkeley Open Infrastructure for Network Computing)[Anderson, 2004] est un intergiciel de traitement distribué pour les GC, développé à l'origine pour le projet *SETI@home* [4] [Anderson et al., 2002] avant qu'il devienne une plateforme générique pour d'autres projets de grille et de calcul distribué. Grace à l'ouverture du projet, la simplicité des outils et la participation des volontaires de partout dans le monde, la plateforme compte à présent plus que 400 000 utilisateurs actifs et 600 000 ordinateurs connectés [Giorgino et al., 2010].

Après l'installation d'un client BOINC et le choix d'un projet (ex. *Climate@home*, *Einstein@home*, *Asteroids@home*, *PrimeGrid*, *ATLAS@home*, etc.), l'application reçoit un ensemble de tâches depuis le serveur ordonnanceur du projet.

Les tâches affectées à l'ordinateur du client volontaire dépendent principalement des performances de ce dernier. Par exemple, le serveur n'affecte pas une tâche qui demande plus de mémoire vive que la mémoire disponible. L'affectation des tâches s'effectue d'une façon aléatoire tant que les performances requises sont disponibles. Une fois les tâches ordonnancées sur l'hôte client, le téléchargement des données d'entrée à partir du serveur de données, le traitement, et la production des données de sortie démarrent. Les fichiers de sortie associés aux tâches accomplies sont renvoyés au serveur de données. Lorsque toutes les tâches ordonnancées sont achevées, l'application envoie un rapport au serveur ordonnanceur qui affectera ou pas (selon les préférences du client) des nouvelles tâches, et le cycle reprend.

La plateforme BOINC est gérée par un serveur central. Chaque client qui se connecte à ce serveur télécharge les composants logiciels nécessaires ainsi que les données de calcul. Le traitement de ces données s'effectue d'une manière distribuée. Néanmoins, les tâches de gestion de la grille ainsi que

2. http://www.distributedcomputing.info/projects.html
3. http://boinc.berkeley.edu/
4. http://setiathome.ssl.berkeley.edu/

la distribution des fichiers de données sont encore centralisés pour des raisons de sécurité. Dans [Farkas et al., 2011], Farkas propose la transformation de l'architecture de BOINC en une architecture distribuée basée sur un système P2P afin de diminuer le flux de données géré par le serveur central. La technologie du P2P, tel que le protocole BitTorrent [Han et al., 2014, Pouwelse et al., 2005, Zhang et al., 2010, Zhang et al., 2009], est aussi envisagée pour être intégrée dans BOINC afin de distribuer la duplication des fichiers de données [Anderson, 2004].

#### II.2.3.3 Informatique dans les Nuages, "Cloud Computing"

Le paradigme informatique dans les nuages ou "Cloud Computing" (CC) a immergé récemment le milieu industriel [Armbrust et al., 2009, Armbrust et al., 2010]. Le CC traduit la volonté de transformer les ressources de traitement et de stockage en une utilité quantifiable afin de l'intégrer dans le marché des TIC. Les termes "on-demand computing" (informatique à la demande), "software as a service" (logiciel en tant que service), "Internet as a Platform" (Internet en tant que plateforme) sont d'autres désignations du CC [Hayes, 2008]. Selon le rapport technique de l'université de Berkeley élaboré en 2009 [Armbrust et al., 2009], le paradigme de l'informatique dans les nuages se réfère simultanément aux applications fournies comme étant un service à travers Internet et aux ressources matérielles et logicielles des Datacenters qui fournissent ces services.

Le CC n'a pas de définition unique. Néanmoins, plusieurs chercheurs dans ce domaine se sont mis d'accord sur le fait que ce concept est le dérivé et la composition des trois paradigmes SaaS, PaaS et IaaS [Armbrust et al., 2009, Armbrust et al., 2010, Fernando et al., 2013, Khan et al., 2013] :
- *SaaS* : "Software as a Service" ou logiciel en tant que service ;
- *PaaS* : "Platform as a Service" ou plateforme en tant que service ;
- *IaaS* : "Infrastructure as a Service" ou infrastructure en tant que service.

##### a. Notions et Définitions

- *Informatique dans les Nuages (CC)* : Selon [Armbrust et al., 2009], le CC est défini comme étant la combinaison des applications fournies comme étant un service à travers Internet ainsi que les ressources matérielles et logicielles des Datacenters qui fournissent ces services. La livraison d'un service de CC peut être effectuée selon 3 modes : SaaS, PaaS et IaaS.
- *Logiciel en tant que Service (SaaS)* : La SaaS [Liu et al., 2009, Marinescu, 2013] est définie comme étant un modèle de présentation des services. En comparaison avec les modèles conventionnels de déploiement des logiciels, les applications SaaS sont stockées dans le réseau du fournisseur et fournies en tant que web services. A travers des interfaces d'abstraction des services, ces applications

sont délivrées à la demande d'une façon simple et efficace aux utilisateurs à travers Internet. Les applications du modèle SaaS sont accessibles à partir d'une variété de terminaux clients à travers une interface de type client-léger comme par exemple un explorateur Internet. L'utilisateur n'est pas autorisé à gérer ni à contrôler l'infrastructure cloud sous-jacente, y compris le réseau, les ressources de traitement et de stockage et les systèmes d'exploitation.

– *Plateforme en tant que Service (PaaS)* : La PaaS [Marinescu, 2013] permet de déployer des applications créées ou acquises par le client en utilisant des outils supportés par l'infrastructure cloud du fournisseur. La gestion et le contrôle de l'infrastructure du cloud (ex. réseau, systèmes d'exploitations et ressources de traitement et de stockage) sont transparents vis-à-vis de l'utilisateur. En effet, l'utilisateur possède le contrôle des applications déployées ainsi que de quelques paramètres de configuration de l'environnement d'hébergement de ces applications.

– *Infrastructure en tant que Service (IaaS)* : L'IaaS [Valencio et al., 2013, Nguyen et al., 2013, Marinescu, 2013] est une notion introduite dans le but de répondre aux nouvelles exigences de gestion des réseaux (ex. protocoles de routage et la virtualisation de l'architecture du système). L'IaaS fournit une représentation abstraite d'un système matériel (ex. ressources de traitement et de stockage, réseau, etc.) créé à partir de l'ensemble des ressources disponibles. Le fournisseur d'une IaaS délivre un ensemble de ressources de traitement et de stockage virtualisées où l'utilisateur peut déployer ses applications et ses logiciels tout en gardant la capacité à gérer et contrôler les ressources de traitement et de stockage disponibles ainsi que les systèmes d'exploitation (i.e. mise à jour, configuration, installation et désinstallation, etc.).

La Fig. II.3 est une représentation simplifiée d'un CC résumant les définitions présentées ci-dessus. Chaque niveau de cette figure correspond à un type de service de CC.

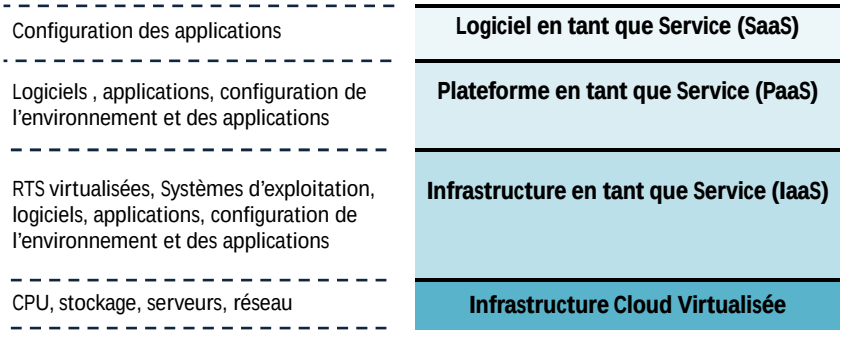



FIGURE II.3: Représentation simplifiée des modes de livraison d'un service de CC. Le niveau d'usage du service CC est spécifiée pour chaque mode.

**b. Le Nuage Mobile, "Mobile CC" (MCC)**

Il existe plusieurs définitions de MCC. Plusieurs recherches ont donné naissance à des concepts différents. Les MCC peuvent être regroupés en quatre catégories comme suit :

– *Mode A.* (Fig. II.4). Dans les travaux de [Cai et al., 2013, Abdo et al., 2014, Kotwal et Singh, 2012], le MCC est introduit de la même manière que les applications populaires Google Docs [5], Microsoft Office Online [6], Dropbox [7], Apple iCloud [8], SoundCloud [9], etc. Dans Google Docs par exemple, l'application cloud consiste en un logiciel de bureautique incluant plusieurs applications telle qu'une application de traitement de texte et un créateur de présentations. L'application est installée sur les serveurs de Google et peut être exécutée à partir de différents appareils mobiles à distance en se basant sur un modèle de type client léger. Comme le montre la Fig. II.4, le développement d'un MCC dans ce cas, revient à développer des applications cloud capables d'être gérées en utilisant des interfaces simples et légères à distance.

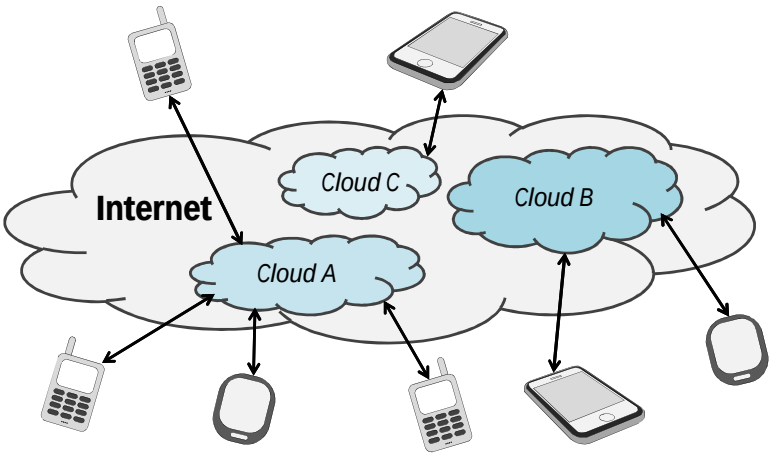



FIGURE II.4: Concept d'interaction Mobiles-Nuages distants en utilisant Internet

– *Mode B.* (Fig. II.5). Dans ce mode de MCC, l'approche consiste à considérer que les TM sont eux mêmes des ressources de traitement et de stockage en se basant sur un système P2P de type MANET (Mobile Ad-hoc NETwork) comme dans [Ku et al., 2014]. Le nuage consiste à virtualiser les ressources de traitement et de stockage d'un ensemble de TM voisins afin de créer un réseau de ressources de traitement et de stockage mobile et momentané plus puissant capable d'exécuter des tâches plus complexes. Dans ce mode de MCC, la gestion des ressources virtualisées s'effectue

5. http://www.google.com/docs/about/
6. http://office.microsoft.com/fr-fr/online/
7. https://www.dropbox.com/
8. https://www.apple.com/icloud/
9. https://soundcloud.com/

en communicant avec une entité centrale de gestion qui permet aux TM qui composent le MCC d'exploiter ce cloud.

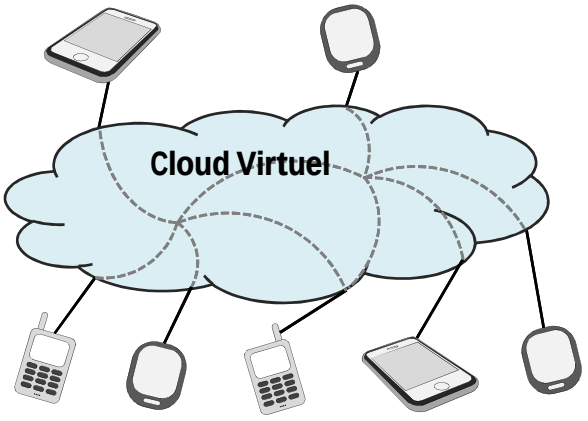



FIGURE II.5: Concept de nuage virtuel créé à partir d'un groupe de mobiles voisins

– *Mode C.* (Fig. II.6). Dans ce mode, un TM se trouvant hors de portée du cloud auquel il est inscrit, peut passer par une entité appelée Cloudlet [Satyanarayanan et al., 2009, Simanta et al., 2012]. Les espaces publics selon ce modèle de cloud seront équipés avec des Cloudlets qui faciliteront la communication avec les serveurs du cloud. Un Cloudlet se comporte dans ce cas comme un point d'accès dans un espace ubiquitaire. L'intérêt de l'usage des Cloudlet est de remédier au problème d'inaccessibilité (ex. hors de portée d'une connexion WiFi) ou de temps de réponse élevé causé par la communication directe avec un cloud. Grâce à une connexion haut débit reliant les Cloudlets aux clouds, les TM seront capables de communiquer avec un cloud tout en gardant des temps de réponses acceptables.

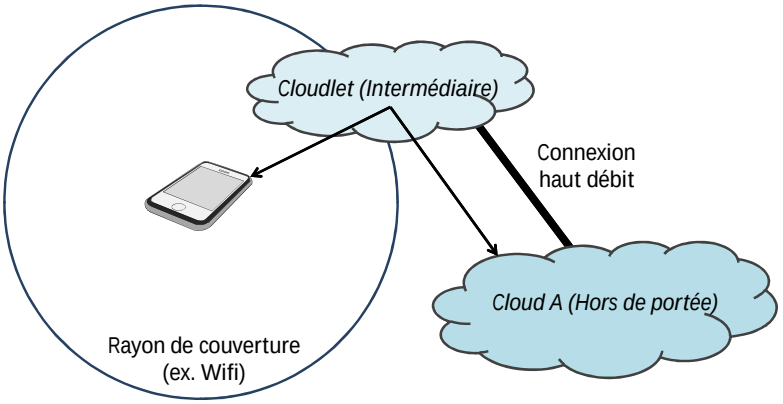



FIGURE II.6: Concept à borne de relais basée sur un cloud intermédiaire

– *Mode D.* (Fig. II.7). Dans ce modèle [Alshareef et Grigoras, 2014], le principe de base ressemble à celui introduit dans le mode C, mais au lieu de définir une entité spéciale pour faire le lien entre un

cloud et un client incapable de l'atteindre, le lien est effectué à l'aide d'un autre client en se basant sur une plateforme de communication de type MANET I.4.3.2.

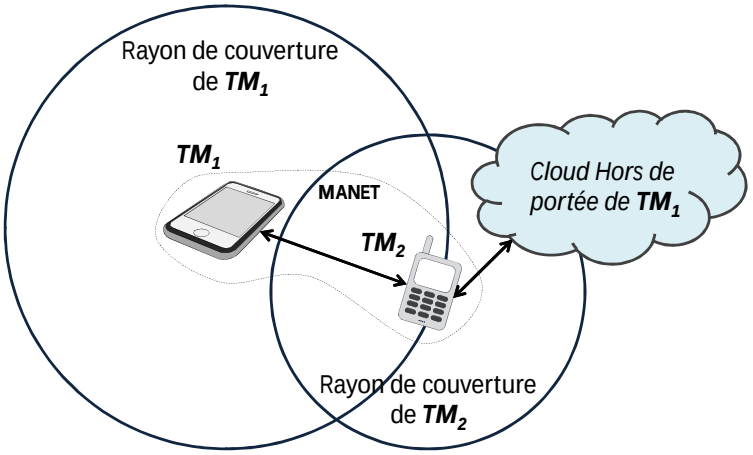



FIGURE II.7: Concept à borne de relais basée sur un TM intermédiaire

Les propriétés des trois modes B, C et D montrent une analogie avec le système proposé dans cette thèse. La différence entre les deux systèmes réside dans fait que le premier se situe au niveau des couches réseau (numéro 3) et transport (numéro 4) du modèle OSI [10]. Par contre, notre système se situe au niveau de la couche d'application (numéro 7) du modèle OSI, donc à une couche plus élevée. Le tableau suivant présente les éléments de l'analogie entre les deux systèmes :

| **Systèmes** | **Modes B, C et D (couches 3 et 4 du modèle OSI)** | **PRoSAM (couche 7 du modèle OSI)** |
|---|---|---|
| **Composants** | nuage virtuel du mode B | voisinage collaboratif |
| | cloudlet | connecteur |

Dans le paragraphe suivant, nous présentons les différentes propriétés et caractéristiques des systèmes multi-agents (SMA). Nous introduisons ensuite des notions d'optimisation génétique et nous présentons l'apport de l'alliance SMA-optimisation dans le cas de nos travaux.

## II.3 Modélisation et Développement Orientés Agent

Jusqu'à ce stade, les caractéristiques et les spécifications des systèmes d'informations étendus sont présentés à travers une revue détaillée de la littérature. La modélisation des systèmes collaboratifs distribués et étendus basée sur le paradigme agent est présenté dans ce paragraphe. En particulier, la définition et les propriétés du paradigme agent et du système multi-agent ainsi que la notion d'agent mobile sont aussi présentées.

10. Open Systems Interconnection

### II.3.1 Intelligence Artificielle Distribuée et Paradigme Agent

Il y a vingt ans, les Systèmes Multi-Agent (SMA) ont émergé dans le domaine de l'IA. De nos jours, les SMA ne sont plus des simples sujets de recherche mais aussi un sujet d'enseignement académique et d'applications industrielles et commerciales. Dans ce paragraphe, nous présentons l'intelligence artificielle distribuée (IAD) en tant que l'origine de la naissance des SMA. Nous présentons ce que nous entendons par agent ainsi les spécifications de ce dernier. Nous nous focalisons en particulier sur les agents logiciels.

#### II.3.1.1 Intelligence Artificielle Distribuée

L'intelligence artificielle distribuée (IAD) est définie comme étant l'étude de la façon dont un ensemble d'agents individuels peuvent combiner leurs ressources afin de résoudre un problème global. L'approche habituelle consiste à diviser le problème original en un ensemble de problèmes plus simples et de résoudre ces derniers indépendamment [Huhns, 2012]. Selon [Gaiti, 1993], dans un système d'IAD, chaque agent utilise des ressources locales et communique avec des agents distants. Bien que le concept de l'IAD a été proposé au départ pour répondre au problème de dispersion des ressources et des unités de traitement [Nilsson, 1981], dans [Ferber et Perrot, 1995], l'auteur propose une réponse plus généralisée à la question "pourquoi distribuer l'intelligence ?" et justifie le recourt à l'IA distribuée ou collective pour apporter des solutions informatiques intelligentes à un problème caractérisé par les contraintes suivantes :

- le problème soit distribué géographiquement. Ceci peut être le cas d'un système de contrôle d'une chaîne logistique ou d'un système de vote électronique national par exemple ;
- le problème fasse intervenir des ressources hétérogènes et distribuées. Ceci nécessite la participation de différents acteurs de différentes spécialités. La coopération du groupe dans ce cas nécessite un système multi-spécialités capable de comprendre les différents membres du groupe et de réussir la coopération en faveur de l'objectif global ;
- penser ouvert et concevoir des systèmes ouverts [11] soit une nécessité émergente. Les systèmes fermés ne répondent plus aux exigences par rapport à l'interopérabilité. De plus, ces systèmes n'exploitent pas pleinement les capacités offertes par Internet ;
- le problème soit suffisamment complexe pour ne pas pouvoir être résolu avec une vision globale. De nouvelles approches de conception des outils informatiques s'introduisent de plus en plus telle que la conception kénétique [Ferber et Perrot, 1995]. Ce paradigme consiste en la conception de logiciels

11. Nous entendons par un système ouvert tout système en interaction continue avec des utilisateurs et/ou avec d'autres systèmes

capables de s'adapter et évoluer par interaction avec d'autres composants de leur environnement physiquement distribué. En d'autres termes, des logiciels avec des visions locales seront capables de résoudre un problème global en interagissant les uns avec les autres ;
- la solution apportée doit répondre aux nouvelles exigences d'adaptabilité et d'évolution pour maintenir l'efficacité et la robustesse au sein d'un environnement dynamique et en évolution continue ;
- la tendance actuelle de conception des systèmes informatiques consiste en l'adoption accrue d'entités autonomes et communicantes capables de coexister dans un environnement distribué. D'ailleurs, nous entendons parler de plus en plus de la programmation "orientée agent" comme étant le successeur de la programmation "orientée objet".

La distribution de l'IA peut être envisagée sous plusieurs formes différentes. Selon [Bond et Gasser, 1988] il existe cinq types de distribution :

- *Distribution spatiale ou géographique* : dans ce type de distribution, le système est composé d'un ensemble d'entité dispersées géographiquement et utilisant un matériel distribué (ex. des véhicules distants). Ceci permet au système d'avoir des entrées (ex. capteurs séismiques, caméras de surveillance, télescopes astronomiques, etc.), des sorties (ex. actionneurs, modules d'alertes, bras robotiques, etc.), des bases de données, etc. spatialement distribuées ;
- *Coût de traitement* : l'emplacement d'une information ou d'une compétence spécialisée peut être un facteur considérable pour définir son coût. Par conséquent, le coût des informations peut être considéré comme un critère de distribution ;
- *Distance sémantique* : les connaissances sont classées selon des catégories et des thématiques afin de permettre l'utilisation pratique et efficace plus tard (ex. informations linguistiques, mathématiques, physiques, historiques, etc.). Il existe plusieurs formes de corrélation entre les classes de connaissances comme des classes peuvent hériter d'autres classes. Cette multitude de classes peut être à l'origine de la distribution ;
- *Distribution temporelle* : dans ce type de distribution, un système d'IAD est défini en termes de coûts des évènements et des connaissances qui peuvent apparaître et disparaître au cours du temps. Ces informations distribuées dans le temps, peuvent être déduites à travers le temps en utilisant des systèmes avec des capacités d'apprentissage ;
- *Distribution logique* : cette distribution est basée sur le niveau de dépendance logique entre des éléments de connaissance. En d'autres termes, la production de nouvelles connaissances se base sur un processus intermédiaire de déduction ou d'inférence qui définit, entre autres, la distance logique entre les éléments de la connaissance.

Le type de distribution choisi dans un système d'IAD permet de définir les spécifications et les compétences des agents qui composent le système, ce qu'on appelle le processus "d'agentification" [Pujalte-Busseuil et al., 2006, Sordoni et al., 2009]. Dans ce qui suit, nous introduisons les caractéristiques du paradigme agent.

#### II.3.1.2 Paradigme Agent

Selon [Woolridge, 2001], un agent est un système informatique situé dans un environnement et capable d'exécuter des actions d'une manière autonome pour atteindre ses objectifs. Selon [Shoham, 1997], une définition plus sociale est proposée : un agent est considéré comme étant une entité composée d'un ensemble de composantes mentales telles que des croyances, des capacités, des choix et des engagements. Selon [Jennings et Wooldridge, 1996] un agent intelligent possède quatre principales caractéristiques :

- Autonomie : les agents sont capables d'exécuter la plupart de leurs tâches sans l'intervention de l'être humain ou d'autres agents et possèdent un certain degré de contrôle de leurs actions et de leurs états internes ;
- Sociabilité : les agents sont capables d'interagir au bon moment avec d'autres agents ou êtres humains pour compléter leurs tâches ou aider les autres agents ;
- Réactivité : Les agents doivent percevoir leur environnement et répondre aux changements qui s'y produisent ;
- Proactivité : En plus de la réponse aux stimulations de l'environnement, les agents doivent être opportunistes, avoir un comportement dirigé par un objectif et prendre l'initiative quand il le faut.

Situé dans un environnement réel ou virtuel, un agent peut jouer un ou plusieurs rôles dans une organisation. Ses motivations sont générées par ses propres objectifs qu'il cherche à satisfaire continuellement. L'agent peut être capable de communiquer avec d'autres agents. Il peut agir sous la conduite d'un autre utilisateur ou d'un autre agent, anticiper, apprendre de ses expériences et s'adapter à son environnement.

En présence d'autres agents, deux cas de figure se présentent :

- les agents coordonnent leurs actions afin d'éviter les conflits de ressources pour atteindre collectivement le même objectif ;
- les agent possèdent des objectifs différents et négocient pour gérer les situations de conflit.

#### II.3.1.3 Interactions Inter-Agents

Dans un système multi-agent, un ensemble d'agents est en interaction permanente les uns avec les autres afin de résoudre un problème bien défini. Dans le cas général, les agents agissent aux noms des utilisateurs avec différents objectifs et motivations [Wooldridge, 2009]. Afin de réussir leurs interactions, ils ont besoin de la capacité de coopérer, de coordonner et de négocier les uns avec les autres. L'aspect organisationnel et social des SMA traduit par les aspects de coopération, de coordination et de négociation, représente le plus grand apport de ces systèmes dans le domaine des sciences de l'informatique.

– La coopération (Fig. II.8) : les agents coopèrent lorsqu'ils sont déterminés à réaliser un objectif global. Dans ce cas on procède à une technique de répartition des tâches appelée coopération ou collaboration ;

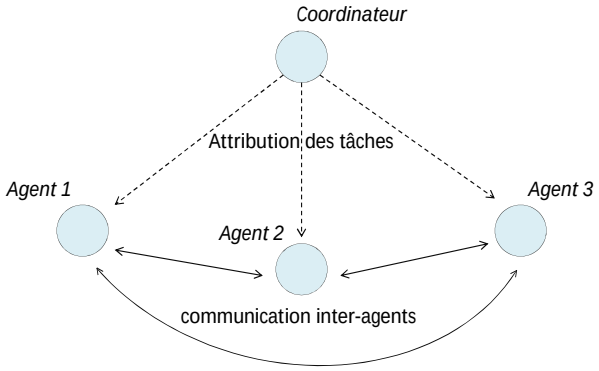



FIGURE II.8: Coopération

– La coordination (Fig. II.9) : c'est la gestion des interdépendances entre les activités des agents. C'est un mécanisme adopté par les agents coopérants pour pouvoir réussir leur coopération ;

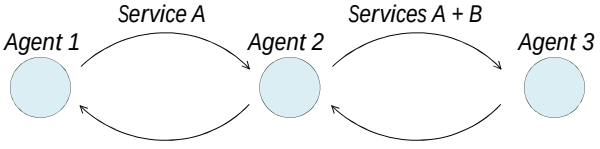



FIGURE II.9: Coordination

– La négociation (Fig. II.10) : c'est le cas où chaque agent est motivé individuellement pour atteindre son propre objectif, en dépit de ceux des autres agents, mais en essayant de conserver certaines propriétés au niveau du groupe.

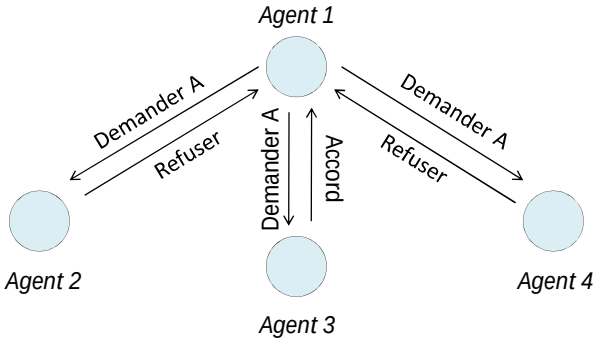



FIGURE II.10: Négociation

#### II.3.1.4 Discussion

En considérant la problématique traitée dans cette thèse, les SMA représentent la solution appropriée pour résoudre des problèmes distribués. Leur architecture distribuée basée principalement sur une vision locale du problème ainsi que leurs caractéristiques intrinsèques d'autonomie, de sociabilité, de réactivité et de proactivité, leur permettent de convenir aux besoins d'évolutivité et d'adaptabilité. Les SMA sont des outils efficaces et robustes pour construire des architectures ouvertes, distribuées, hétérogènes, flexibles et capables de résoudre les problèmes en coopérant et coordonnant leurs actions.

Les systèmes multi-agents issus de l'intelligence artificielle distribuée sont utilisés pour
– concevoir des solutions distribuées pour des problèmes distribués ;
– modéliser les systèmes distribués et développer des outils logiciels distribués ;
– simuler des phénomènes et analyser des comportements émergents ;
– etc.

Ces systèmes sont critiqués par la communauté des systèmes distribués (présentés précédemment dans le paragraphe II.2.3) et accusés de ne pas apporter des contributions significatives à la problématique de modélisation et de simulation des problèmes distribués. Le paragraphe suivant présente les différences et les similarités entre les systèmes distribués et les systèmes multi-agents et démontre l'intérêt de ces derniers dans le cadre de cette thèse.

### II.3.2 Systèmes Distribués et Systèmes Multi-Agents

Selon Ferber [Ferber et Perrot, 1995], il existe une grande différence entre les systèmes distribués et les systèmes multi-agents (SMA). En effet, "(...) les systèmes distribués cherchent à concevoir des systèmes informatiques capables de gérer l'exécution de tâches en employant au mieux des ressources

physiquement réparties (mémoire, processeur, gestionnaires d'informations). Les techniques mises en œuvre (client/serveur, distribution des applications) relèvent plus spécifiquement de la pratique informatique ou, plus exactement, d'une opérationnalisation de systèmes informatiques".

La problématique principale selon lui, consiste à "(...) concevoir, spécifier et valider des applications réparties, en définissant des protocoles permettant à plusieurs modules informatiques répartis d'utiliser mutuellement leurs services". Contrairement à la problématique traitée dans les systèmes distribués qui s'avère orientée application et implémentation, la problématique des SMA se montre beaucoup plus abstraite et s'intéresse à des problématiques plus ouvertes telles que :

– l'autonomie, la sociabilité, la réactivité et la proactivité d'un agent ;
– la coopération, la coordination et la négociation en cas de conflit d'intérêts dans un système d'agents.

Ferber affirme que "(...) la perspective de type "résolution de problèmes", "coordination d'actions" ou "partage de connaissances" que soulèvent les SMA ne correspond pas du tout aux systèmes distribués". Néanmoins, il est important de souligner que les recherches menées dans le domaine des SMA ne sont pas parties de zéro. En effet, "(...) les SMA utilisent un grand nombre de techniques provenant des systèmes distribués, en se situant à un niveau plus conceptuel et méthodologique".

Ferber conclut ses propos en soulignant la complémentarité entre les deux domaines, qui relève plutôt une synergie qu'une véritable opposition : les systèmes distribués apportent "(...) leur rigueur et leurs algorithmes", les SMA donnent "une conceptualisation et une approche plus générale, moins centrée sur les mécanismes d'exécution".

Le contexte purement applicatif et technologique (les espaces ubiquitaires) ainsi que l'approche de résolution proposée (approche multi-agent) et traitée par la problématique de la thèse exigent la prise en considération des paramètres concrets de l'environnement (services, clients, fournisseurs de services, requêtes clients, terminaux clients, temps de réponse, coût, délai d'exécution, etc.). Ce contexte impose aussi la mise en œuvre des modèles (macro et micro) plus abstraits qui prennent en compte les différents types d'interactions entre les composants du système (les agents) en terme de coopération, coordination et négociation. Nous pensons que les SMA sont capables de résoudre notre problème qui est un problème distribué. Nous admettons aussi que les SMA représentent un outil convenable d'abstraction du problème traité dans cette thèse.

Les SMA introduisent une type particulier d'agent : l'agent mobile. Cette notion est introduite pour remplacer les anciennes solutions de communication dans le réseau qui sont basées sur des RPC [12]. La définition du paradigme agent mobile est présentée dans ce qui suit.

12. "Remote Procedure Call" : appel de procédure distante

### II.3.3 Paradigme Agent Mobile

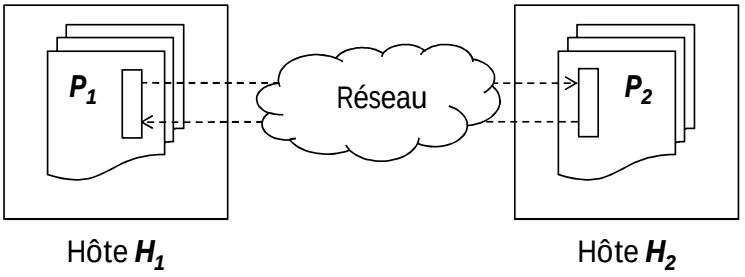
Les *agents mobiles* sont des agents capables de se transmettre eux même, i.e. leur programme (les comportements et les méthodes) et leurs états internes (les variables et les constantes), à travers le réseau et ensuite reprendre l'exécution des tâches sur un hôte distant.

L'introduction des agents mobiles a pour but de remplacer les RPC (Remote Procedure Call) et proposer un nouveau concept de communication à travers le réseau. Dans les RPC, l'idée consiste à faire un appel d'une méthode d'un processus $P_2$ depuis un processus distant $P_1$. D'une part, cet appel s'effectue d'une façon *synchronisée* ce qui implique que le processus appelant se bloque tout au long de l'exécution de la demande par le processus appelé jusqu'à ce que ce dernier renvoie le résultat. Ceci peut être dangereux dans un système sensible (ex. un réseau de capteurs de détection des ondes séismiques) où un échec du support de communication peut entrainer un blocage infini du processus appelant (par exemple, ceci peut être résolu en utilisant un délai d'attente maximale, un *timeout*). D'autre part, la connexion réseau entre les deux processus peut être maintenue, néanmoins, la mal-exploitation du réseau peut être coûteuse.

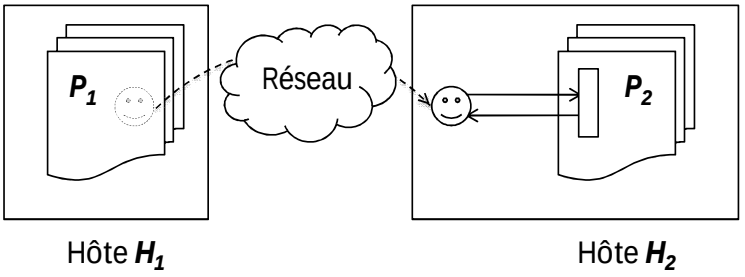
Par conséquent, le concept d'agents mobiles consiste à envoyer un agent dans un hôte distant afin d'y effectuer des traitements et/ou exécuter des instructions. Ainsi, au lieu d'appeler une méthode du processus $P_2$, le processus $P_1$ transmet un programme qui s'exécutera dans l'hôte de $P_2$. Dans ce cas, l'agent mobile est capable d'exécuter ses tâches d'une façon plus efficace lorsqu'il se situe sur la même hôte que le processus appelé que lorsqu'il utilise le réseau pour envoyer des requêtes et recevoir des réponses. La Fig. II.11 présente une illustration des deux modèles d'interaction à distance : rpc et agent mobile.

D'après [Zgaya, 2007], le paradigme agent mobile utilisé pour collecter des données à partir des nœuds du réseau peut démontrer des performances meilleures que celles du paradigme classique client/serveur. En effet, en faisant varier la quantité de données transférées à chaque fois en utilisant les deux paradigmes : agent mobile (AM) et client/serveur (CS), l'AM montre une meilleure performance lorsque la taille des données devient importante (Fig. II.12).

Lors de la conception d'un système à base d'agents mobiles, il existe quelques questions qui se posent, telles que les questions de sécurité (ex. sécurité des informations situées dans l'hôte qui héberge un agent mobile), de transmission de données (i.e. comment et à quel coût un agent sera capable de se déplacer) ou d'hébergement et d'exécution distante (i.e. comment et à quel coût un agent va être hébergé et exécuté dans un nœud distant du réseau). Ces questions représentent encore des sujets de




(a) Modèle RPC




(b) Modèle Agent Mobile

FIGURE II.11: Illustration des concepts RPC et Agent Mobile

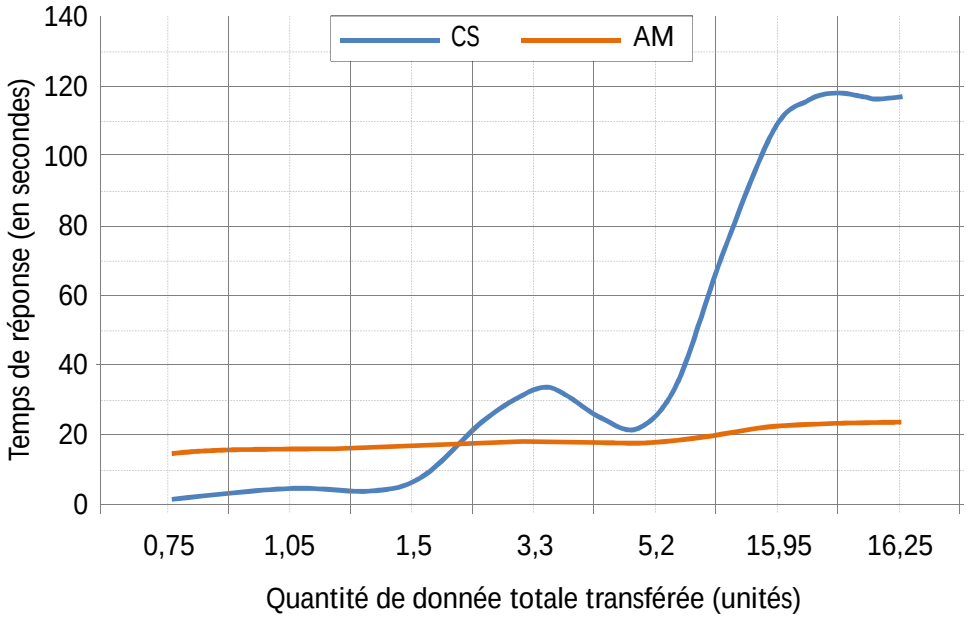



FIGURE II.12: Comparaison des performances des paradigmes Agent Mobile (AM) et client/serveur (CS).

recherche au sein de la communauté des SMA mais le travail présenté dans cette thèse ne s'intéresse pas à cette problématique particulière.

Le paradigme "rôle" dans les SMA permet d'élaborer des modèles et des solutions d'une manière abstraite ce qui permet d'avoir des solutions plus généralisées et simplifie en même temps la tâche d'implémentation. Dans ce qui suit, nous introduisons le paradigme "rôle" dans les système multi-agents. Nous présentons en particulier la définition du rôle et les travaux élaborés au tour de cette notion dans les systèmes multi-agents.

### II.3.4 Paradigme Rôle dans les Systèmes Multi-Agents

Les SMA sont basés sur les notions de société et d'organisation. Généralement, une organisation ou une société est composée de plusieurs individus qui possèdent chacun un rôle ou une fonction (plusieurs individus peuvent avoir le même rôle et plusieurs rôles peuvent être affectés à un seul individu) qui permet de définir leurs droits et leurs devoirs vis-à-vis du reste des individus. Par conséquent, il est intéressant d'utiliser cette notion de rôle dans une organisation pour modéliser un système d'agents. Dans le paragraphe suivant, nous présentons la définition de ce concept de point de vue organisation artificielle.

#### II.3.4.1 Qu'est ce qu'un Rôle ?

Le concept de rôle comme étant un outil de modélisation a été étudié depuis l'année 2000 [Wooldridge et al., 2000]. D'autres études plus récentes telles que [Adam et Mandiau, 2007, Zhu et Zhou, 2008, Durmus et Erdogan, 2009, Zhu et al., 2010, Zhu et Zhou, 2010, Ferrari et Zhu, 2011] montrent que ce paradigme est convenable pour la modélisation des organisations, des interactions au sein des groupes, des protocoles de communication, des systèmes hiérarchiques des organisations, des dépendances entre les composants et des droits d'accès aux ressources.

Ainsi, le rôle peut être défini comme étant une collection de facultés et de comportements acquis par l'individu au sein de l'organisation. Il peut être représenté par l'objectif de l'agent (ex : quand un agent a pour objectif de vendre un service $s_1$, dans ce cas il a le rôle d'un fournisseur, mais si son objectif est d'acheter un service $s_2$ alors il a le rôle d'un client). Les agents peuvent ainsi acquérir et perdre des rôles lorsque leurs objectifs changent [Durmus et Erdogan, 2009]. Les rôles sont utilisés dans plusieurs contextes comme étant un outil :

– d'abstraction, d'analyse et de design des SMA [Wooldridge et al., 2000] ;
– de résolution des problèmes de collaboration et de gestion des conflits dans les systèmes administratifs complexes [Zhu et al., 2010][Zhu et Zhou, 2010] ;
– d'assistance de la collaboration Homme-Machine et Machine-Machine ;
– de modélisation des architectures holoniques (structures hiérarchiques basées sur des agents holoniques capables de créer des clones, leur déléguer des tâches et les superviser) pour l'automatisation des processus Workflow [Adam et Mandiau, 2007].

Le paradigme rôle est un concept de modélisation intéressant dans les SMA. D'une façon plus générale, il peut être utilisé pour modéliser les organisations, les interactions au sein des groupes, les protocoles de communication, les systèmes hiérarchiques des organisations, les dépendances entre les composants, les droits d'accès aux ressources, etc. Dans le paragraphe suivant nous présentons quelques travaux de recherche dans le domaine de la modélisation orientée rôle des SMA.

#### II.3.4.2 Modélisation orientée rôle dans les SMA

Initialement, l'utilisation du paradigme rôle pour modéliser une organisation à base d'agents logiciels a été intuitive et n'a pas eu une définition standard [Zhu et Zhou, 2008]. Chaque travail définit le rôle selon ses besoins (de modélisation ou d'implémentation). Ainsi, le paradigme rôle évolue sur deux axes parallèles :

– la modélisation des organisations artificielles en se basant sur les services et les méthodes qu'un agent peut fournir au sein du groupe. Ceci est le cas des systèmes de type CSCW (Computer-Supported Cooperative Work) ;
– la sécurité dans les systèmes informatiques en se basant sur les droits d'accès aux ressources et les demandes provenant de différents utilisateurs. Ceci correspond au modèles de type RBAC (Role-Based Access Control).

Dans [Ferrari et Zhu, 2011], les auteurs présentent leur implémentation du rôle en créant un framework de simulation appelé WhiteCat. Les propriétés les plus intéressantes de ce framework sont le dynamisme et la perceptibilité :

– Dynamisme : pouvoir changer le rôle dynamiquement pendant l'exécution ;
– Perceptibilité : un agent peut percevoir les rôles des autres agents uniquement en les "regardant". En effet, le rôle est inscrit dans l'entête de la classe de l'agent et se modifie pendant le temps d'exécution.

Dans [Durmus et Erdogan, 2009], les auteurs proposent une architecture AWSM (Agent Web Service Market) pour la présentation, la vente et l'achat des web services. Dans cette étude, nous remarquons une absence des mécanismes d'évaluation des agents et de changement dynamique des rôles. Le formalisme proposé est restreint, il ne prend pas en compte les ressources par exemple. Dans [Adam et Mandiau, 2007], une architecture d'agents holoniques a été proposée pour la recherche et la récupération des documents depuis Internet. Cette architecture est composée de cinq types d'agents :

- agent coordinateur : responsable de la coordination avec les autres groupes d'agents du système ;
- agent interface : récupère la requête client et l'enregistre dans une base de données ;
- agent d'information : consulte continuellement la base de données afin de vérifier les nouvelles demandes de recherche ;
- agent requête : responsable de la classification sémantique des requêtes utilisateurs ;
- agent de recherche : chaque agent de ce type est spécialisé dans un domaine sémantique différent et responsable de répondre à la demande client correspondante.

Cette plateforme permet une recherche sémantique qui associe un agent de recherche à chaque domaine sémantique de recherche. La centralisation du système peut être expliquée par le fait que ce type d'architecture est une pure copie artificielle de l'architecture d'une organisation réelle. En effet, l'organisation SMA est basée sur des règles d'interaction issues des propriétés et des modes de fonctionnement des organisations réelles, qui sont souvent basées sur des architectures hiérarchiques centralisées.

#### II.3.4.3 Gaia : Méthodologie Orientée Rôle

La méthodologie Gaia proposée dans [Wooldridge et al., 2000] permet de modéliser des SMA en se basant sur le paradigme rôle. Comme le montre la Fig. II.13, Gaia est composée essentiellement de trois phases :

1. Phase de définition et formalisation des besoins dans le système ;
2. Phase d'analyse : dans cette phase on définit les modèles des rôles et d'interactions ;
3. Phase de design : dans cette phase on définit les modèles d'agents, des services et d'accointances.

Pendant la phase d'analyse, la définition du modèle des rôles consiste à définir les rôles dans une organisation ainsi que les quatre attributs de chaque rôle : ses responsabilités (fonctionnalités, services), ses permissions (droits, ressources accessibles), ses activités (manière de réalisation des responsabilités, méthodes privées) et ses protocoles (manière d'interagir avec les autres rôles, méthodes nécessitant

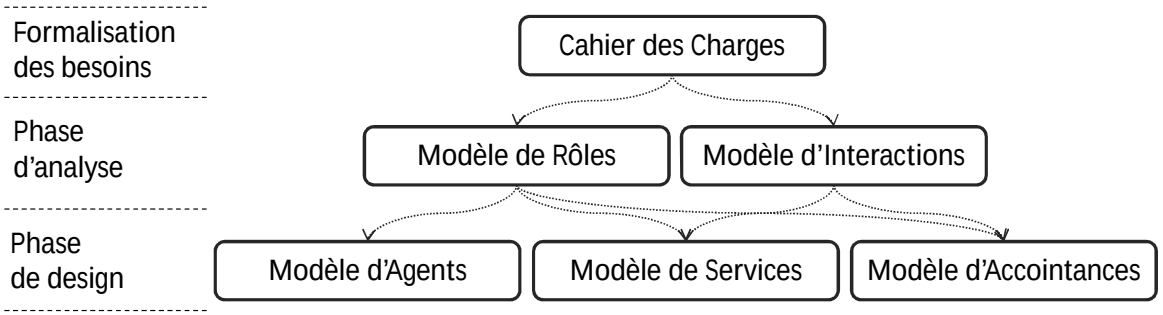



FIGURE II.13: Relations entre les modèles de Gaia

l'interaction avec d'autre agents). La définition du modèle d'interactions consiste à détailler les protocoles d'interaction entre les différents rôles.

La définition d'un protocole repose sur la définition des attributs suivants : la nature ou le but de l'interaction, l'initiateur, le récepteur, les entrées (informations utilisées par l'initiateur), les sorties (informations fournies par le récepteur), le traitement (travail à faire par l'initiateur lors de l'interaction).

Quant à la phase de design, le but est de transformer les modèles abstraits générés lors de la phase d'analyse en modèles suffisamment concrets faciles à implémenter. Pendant cette phase, on définit un modèle d'agents. Ce dernier consiste à définir les correspondances un-à-un entre les rôles et les types d'agents ainsi que le nombre d'agents correspondants à chaque rôle. A ce niveau, un designer peut choisir d'encapsuler un ou plusieurs rôles qui s'avèrent proches en fonctionnalités dans un seul type d'agent.

Dans cette même phase, on définit un modèle de services. Ceci consiste à identifier les services associés à chaque rôle d'agent et spécifier les propriétés principales de ces services. Chaque rôle est associé au moins à un service et chaque service possède un ensemble d'attributs : entrées et sorties (dérivent du modèle d'interactions), pré-conditions et post-conditions (dérivent des contraintes sur les services exprimés par les propriétés de sécurité). La manière dont un agent réalise ces services n'est pas traitée dans Gaia, puisqu'on considère que les services dépendent seulement du domaine d'application.

Enfin, le modèle d'accointances consiste à définir tout simplement les liens de communication qui existent entre les types d'agents. En particulier, il permet d'identifier les éventuels goulots d'étranglement qui peuvent générer des problèmes lors du la mise en œuvre.

La méthodologie Gaia est appropriée pour les SMA qui possèdent les caractéristiques suivantes :

– Chaque agent possède des ressources informatiques considérables ;

– Les agents travaillent en collaboration (i.e. pas de conflits entre les membres de l'organisation) ;
– Les agents sont hétérogènes et peuvent être implémentés par différents langages ;
– La structure organisationnelle du système est statique. En effet, la méthodologie ne supporte pas les changements dynamiques des rôles ou des relations inter-agents ;
– Les comportements des agents ainsi que les services qu'ils fournissent sont statiques et ne changent pas lors du run-time ;
– Le système ne peut supporter qu'un nombre limité de types d'agents (un nombre inférieur à 100).

Cependant, Gaia présente les limitations suivantes :

– n'intègre pas des mécanismes de gestion des conflits ;
– ne supporte pas les architectures dynamiques et enfichables ;
– n'intègre pas des patrons de conception comme en programmation orientée objet ;
– ne supporte pas les protocoles coopératifs qui permettent la négociation ;
– ne prend pas en compte un standard particulier (ex. FIPA ACL ou KQML).

#### II.3.4.4 Synthèse

Les études réalisées autour du paradigme rôle dans les SMA montrent que cette technique est utilisée principalement pour modéliser les organisations réelles et offrir des solutions informatiques qui supportent la collaboration entre les différents éléments de ces organisations. Nous pensons que la notion rôle dans les SMA peut être utilisée différemment. Ce paradigme peut nous permettre de modéliser le comportement dynamique au sein d'un système. Dans ce cas, les agents peuvent changer de rôle d'une manière dynamique ce qui donne une grande flexibilité et extensibilité au système.

Les agents dans un SMA peuvent adopter différents rôles selon l'état du système ainsi que leurs propres objectifs et états internes. De plus, un SMA, regroupant des agents collaboratifs, doit intégrer un système de communication efficace et fiable afin de réussir la collaboration. La communication inter-agent est établie à travers des protocoles de communication qui sont introduits dans le paragraphe IV.6. Les agents évoluant et opérant dans un environnement dynamique, doivent être en mesure de chager leur comportement et probablement réorienter dans un sens différent leurs prises de décisions.

De plus, il est essentiel que les actions de l'agent soient guidées par la recherche de l'optimalité. Par conséquent, les agents intervenant dans cet environnement peuvent intégrer des méthodes d'optimisation adaptées à leurs compétences et connaissances. Optimisation et agents vont ainsi de pair pour réaliser un service optimisé tout en étant efficace et performant.

Nous proposons dans cette thèse un système de gestion des services d'aide à la mobilité basé sur un système d'agents optimisateurs. Nous utilisons des algorithmes génétiques pour l'optimisation des solutions générées par le système. Dans la suite, nous présentons un rappel de la notion d'optimisation mathématique et en particulier l'optimisation évolutionnaire.

## II.4 Optimisation : Vers une Alliance avec un Système d'Agents Intelligents

### II.4.1 Optimisation Mathématique

Un problème d'optimisation mathématique, ou seulement un problème d'optimisation, a la forme suivante :

$$\begin{array}{ll} \text{minimiser} & f_0(X) \\ \text{tel que} & f_i(X) \leq c_i, i \in 1..m \end{array} \tag{II.1}$$

Dans cette expression :

- le vecteur $X = (x_1, ..., x_n)$ représente la variable d'optimisation du problème, appelé aussi vecteur de décision ;
- la fonction $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}$ représente la fonction objectif ou fonction de coût ou encore critère d'optimisation ;
- la fonction $f_i : \mathbb{R}^n \rightarrow \mathbb{R}, i \in 1..m$, représente les fonctions contraintes (des égalités et des inégalités) ;
- les constantes $c_1, ..., c_m$ représentent les limites des contraintes.

Un vecteur $X^*$ est appelé optimal ou solution du problème défini par l'équation II.1 s'il possède la valeur objectif minimale parmi tous les vecteurs qui satisfont les contraintes. C'est à dire : $\forall Z$, tel que $f_1(Z) < c_1, ..., f_m(Z) < c_m$, nous avons $f_0(X^*) \leq f_0(Z)$.

Dans le cas général,il existe des classes ou des familles de problèmes d'optimisation, caractérisées par des formes particulières de fonctions objectifs et contraintes. A titre d'exemple, le problème défini par l'équation II.1 est appelé un problème de programmation linéaire si les fonctions objectif et contrainte $f_0, ..., f_m$ sont linéaires, c'est à dire :

$$f_i(x + \mu y) = f_i(x) + \mu f_i(y), \ i \in [0, m] \tag{II.2}$$

pour $\forall x, y \in \mathbb{R}^n$ et $\forall \mu \in \mathbb{R}$.

Si le problème d'optimisation est non linéaire, nous l'appelons un problème de programmation non linéaire. Dans ce qui suit, la formalisation d'un problème d'optimisation est présentée à travers des exemples pratiques provenant de plusieurs domaines de recherches.

### II.4.2 Conception d'un Problème d'Optimisation

Le problème d'optimisation défini par l'équation II.1 est une abstraction du problème de recherche du meilleur choix d'un vecteur dans $\mathbb{R}^n$ parmi un ensemble de choix candidats. En effet,

- la variable $X$ représente le choix effectué ;
- les contraintes $f_i(X) \leq c_i$ représentent les exigences exprimées dans un cahier de charge ou les spécifications qui limitent les choix possibles
- la valeur objectif $f_0(X)$ représente le coût de choisir X

Une solution du problème d'optimisation II.1 correspond à un choix qui a le coût minimal parmi tous les choix qui satisfont les contraintes.

Dans le cas de l'optimisation d'un système de recherche des services de transport, l'objectif est de trouver le meilleur service possible parmi tous ceux proposés par différents fournisseurs, à différents coûts, QoS et temps de réponse, etc. La variable $x_i$ représente la valeur associée à chaque critère. Dans ce cas, le vecteur de décision $X \in \mathbb{R}^n$ décrit chaque service proposé par le système. Les contraintes de ce problème peuvent être les préférences des utilisateurs associées à chaque critère. Dans ce cas, il existe plusieurs solutions pouvant satisfaire l'utilisateur. La sélection de la meilleure solution est assurée par la fonction objectif qui est responsable de l'évaluation de ces solutions.

Lors de l'optimisation d'un portefeuille par exemple, la recherche est focalisée sur la meilleure méthode pour investir un capital dans un ensemble de $n$ biens. La variable $x_i$ représente l'investissement sur le $i^{\text{ème}}$ bien. Dans ce cas, le vecteur de décision $X \in \mathbb{R}^n$ décrit le portefeuille d'allocation totale à travers l'ensemble des biens. Dans ce contexte, les contraintes peuvent représenter une limite sur le budget (i.e. la somme totale à investir), des exigences sur les investissements pour qu'ils ne soient pas négatifs ou une valeur minimale des revenus totaux du portefeuille. La fonction objectif peut être une évaluation des risques ou des variation des revenus du portefeuille.

Un autre exemple est le dimensionnement des appareils dans la conception électronique. Cette tâche consiste à choisir les dimensions de chaque composant dans un circuit électronique. Les contraintes représentent un ensemble d'exigences d'efficacité, de fabrication et de performance telles que les limites des dimensions d'un appareil imposées par le processus de fabrication, les exigences de temps qui vont

assurer que le circuit peut opérer d'une manière sûre à une certaine vitesse, et les limites de l'espace total du circuit. L'objectif dans le dimensionnement d'un appareil est la puissance totale consommée.

Une grande variété de problèmes pratiques qui font intervenir la prise de décision et qui peuvent être transformés en un problème d'optimisation mathématique (ex. optimisation multicritère). L'optimisation mathématique est utilisée dans plusieurs domaines tels que : la gestion des chaînes logistiques, l'ordonnancement, la conception des réseaux, la régulation dans le domaine du transport, etc. Pour la plupart de ces applications, l'optimisation mathématique représente un outil robuste et performant pour l'aide à la décision au profit l'utilisateur humain : le concepteur, l'opérateur d'un système ou le superviseur d'un processus industriel.

### II.4.3 Optimisation Multi-objectif et Evolutionnaire

#### II.4.3.1 Optimisation Multi-Objctif

Lorsque la modélisation d'un problème d'optimisation présente plusieurs objectifs et nécessite le choix des valeurs de plusieurs variables de décision, il s'agit dans ce cas d'un Problème d'Optimisation Multi-Objectif (POMO). Dans un POMO nous supposons que :

- un nombre de k objectifs doivent être minimisés (ou maximisés) ;
- une solution à ce problème peut être représentée par un vecteur de décision $X = \{x_1, x_2, ..., x_n\}$ dans l'espace de décision $\mathbb{R}^n$ ;
- une fonction $f : \mathbb{R}^n \rightarrow \mathbb{R}^k$ est responsable de l'évaluation de la qualité d'une solution spécifique en l'affectant à un vecteur objectif $Y = \{y_1, y_2, ..., y_k\}$ dans l'espace objectif $\mathbb{R}^k$.

Dans le cas d'un problème à objectif unique (i.e. $k = 1$), une solution $X_1$ est considérée comme étant meilleure qu'une autre solution $X_2$ si la valeur objectif $Y_1 < Y_2$, où $Y_1 = f(X_1)$ et $Y_2 = f(X_2)$. Plusieurs solutions optimales peuvent exister dans l'espace de décision et peuvent être associées à une seule valeur dans l'espace objectif, c'est à dire qu'il n'existe qu'un seul optimum dans l'espace objectif.

Dans le cas d'un POMO (i.e. $k > 1$), la comparaison des solutions est plus complexe. En effet, à cause de l'aspect contradictoire des objectifs (qui est souvent le cas), plusieurs solutions peuvent répondre à un POMO, c'est-à-dire que la diminution d'un objectif peut entraîner une augmentation de l'autre. De ce fait, les solutions obtenues ne seront pas optimales car elles n'optimisent pas tous les objectifs du problème. Dans ce cas, il est propice d'adopter le concept du compromis. Des solutions de compromis optimisent un certain nombre d'objectifs d'un côté, et dégradent les performances des autres objectifs.

La question qui se pose dans ce cas est la suivante : comment choisir une solution convenable parmi une multitude de solutions obtenues ?

En suivant le concept de dominance de *Pareto* [Collette et Siarry, 2002], un vecteur objectif $Y_1$, est dit *dominant* au sens de Pareto par rapport à un autre vecteur $Y_2$ si :

1. aucun des composants de $Y_1$ n'est plus grand que le composant correspondant dans $Y_2$ ;
2. au moins un composant dans $Y_1$ est plus petit que le composant correspondant dans $Y_2$.

La relation de dominance entre $Y_1$ et $Y_2$ est notée comme suit : $Y_1 \prec Y_2$. De la même manière, une solution $X_1$ est considérée comme étant meilleure qu'une autre solution $X_2$ (i.e. $X_1 \prec X_2$) si $Y_1 = f(X_1)$ domine $Y_2 = f(X_2)$.

En effet, les solutions qui ne sont dominées par aucune autre solution peuvent être associées à différents vecteurs objectifs (i.e. plusieurs vecteurs objectifs optimaux peuvent exister et chacun représente des différents compromis entre les objectifs du problème). L'ensemble des solutions optimales de l'espace de décision est en général appelé ensemble de Pareto, notons cet ensemble par $E^*$. L'image de $E^*$ dans l'espace objectif est appelé en général frontière de Pareto que nous notons par $F^*$. Lors de la résolution d'un POMO, la connaissance de cet ensemble permet de contribuer à la prise de décision pour choisir la solution qui donne le meilleur compromis entre les différents objectifs. Dans le paragraphe suivant, les méthodes évolutionnaires, une famille de méthodes réputées et très connues pour la résolution des POMO, sont présentées dans le paragraphe suivant.

#### II.4.3.2 Optimisation Evolutionnaire

La génération de l'ensemble de Pareto peut être coûteuse de point de vue calcul informatique et souvent non réalisable à cause de la complexité de l'application qui empêche les méthodes exactes de trouver une bonne solution en un temps polynomial. Pour cette raison, des stratégies de recherche stochastiques tels que les algorithmes évolutionnaire, la recherche tabou, le recuit simulé et les colonies de fourmis ont été développées. Cette famille de méthodes d'optimisation ne garantit pas souvent l'identification du compromis optimal, mais plutôt d'une approximation. En effet, elle manipule un groupe de solutions réalisables, appelé *population*, à chaque étape du processus de recherche. Cet aspect est inspiré des sciences de la vie et des processus naturels là où les populations d'êtres vivants évoluent selon leurs propriétés collectives et leur environnement. Par conséquence, et par analogie à la science de la vie, l'approche évolutive génère une population initiale, pouvant être créée arbitrairement,

qu'elle fait évoluer le plus naturellement possible dans le but de trouver des bonnes solutions dans l'espace des solutions réalisables.

En général, la taille d'une population reste constante tout au long d'un processus de recherche, qui évolue cycliquement en deux phases se succédant à tour de rôle : une phase de coopération collective entre tous les individus d'une même population (croisement) et une phase d'adaptation individuelle (mutation). Une population courante converge lorsqu'elle contient un pourcentage élevé de solutions identiques. Ce phénomène ne doit pas se produire prématurément afin de laisser assez de temps pour une population d'évoluer, afin de se rapprocher au mieux de la solution optimale. Plusieurs techniques existent pour éviter cet inconvénient, la solution la plus courante étant d'évaluer le degré de diversification d'une population afin de prévoir la diversité et empêcher la convergence précipitée.

En général, les algorithmes de recherche stochastique consistent en trois parties principales :
– un module de stockage des vecteurs de décision considérés comme des candidats de solution ;
– un module de sélection ;
– un module de variation.
L'algorithme 1 illustre les étapes principales d'un algorithme génétique.

**Algorithme 1** Principales étapes d'optimisation génétique

**Étape 1. Initialisation**
- choix du codage
- génération aléatoire de $N$ individus
- initialiser un compteur $i = 1$

**Étape 2. Évaluation**
- évaluer le coût de chaque individu

**Étape 3. Condition d'Arrêt**
- Si la condition d'arrêt est vrai alors Fin

**Étape 4. Sélection**
- Sélectionner $m \leq N$ parents à partir des $N$ individus

**Étape 4. Croisement**
- Créer $m$ enfants à partir des $m$ parents sélectionnés

**Étape 4. Mutation**
- Muter les $m$ enfants avec une probabilité de mutation $p_m$

**Étape 4. Incrémentation**
- $i = i + 1$
- Aller à l'étape 2

Il existe deux types de sélections. Le premier type est une sélection qui s'effectue en choisissant d'une façon aléatoire des solutions potentielles pour les passer au module de variation. Le deuxième type cherche à déterminer lesquelles entre les solutions stockées précédemment et celles qui viennent d'être générées seront gardées dans la mémoire pour les prochaines étapes. Pour la variation, des éléments de l'ensemble des solutions sélectionnées subissent des modifications (automatiques ou aléatoires) dans le

but d'obtenir des meilleures solutions potentielles. Ce cycle sélection-variation peut être répété jusqu'à ce qu'un certain critère d'arrêt soit satisfait.

### II.4.4 Alliance Optimisation-SMA au Profit de la Mobilité Urbaine

Dans cette thèse, nous nous positionnons dans le cadre d'un système fortement sollicité et regroupant des dispositifs et des ressources hétérogènes et très nombreuses. Nous considérons que l'environnement est dynamique et en évolution continue. Ceci implique la nécessité d'optimiser les opérations du système afin de gagner en termes de temps, de coûts et de ressources. L'approche SMA convient parfaitement à ce contexte et est capable de fournir l'efficacité et la robustesse requises dans le cadre d'un environnement dynamique et contraignant. Compte tenu de leurs capacités d'autonomie, de pro-activité, de collaboration au sein d'une organisation, ainsi que d'autres propriétés que nous avons présenté précédemment, les agents sont donc bien appropriés pour représenter le système étudié. Les agents évoluant et opérant dans un tel environnement, doivent pouvoir choisir la bonne pratique à suivre en ajustant leurs comportements et probablement en réorientant dans un sens différent leurs prises de décisions qui vont dépendre directement d'une recherche continue de l'optimalité d'un critère bien défini.

Des études antérieures [Zidi, 2006, Zgaya, 2007, Feki, 2010, Sghaier, 2011] montrent l'efficacité et la justesse de l'association SMA-optimisation. Les agents peuvent intégrer des méthodes et des algorithmes d'optimisation adaptés à leurs objectifs, leurs contraintes et leurs connaissances. L'alliance SMA-optimisation a pour effet de créer deux niveaux d'optimisation : une optimisation locale intégrée dans les actions et comportements des agents, et une optimisation globale gérée par des méthodes d'optimisation distribuées.

Dans [Zidi, 2006], le problème abordé concerne l'optimisation de la recherche des informations correspondant aux requêtes clients dans un Réseau de Transport Multimodal (RETM) distribué par le moyen du paradigme Agent Mobile. Cet agent est capable de se déplacer entre les nœuds du RETM pour traiter les requêtes. La plateforme développée est responsable de l'optimisation de ses plans de route. Dans [Zidi, 2006], la question de mobilité est abordée de point de vue modélisation d'un système de recherche et de composition d'itinéraires en mode normal et la régulation du trafic en mode perturbé. Dans [Feki, 2010], un système distribué d'aide à la décision est proposé pour assister les voyageurs par des plans de déplacement pour constituer un carnet de voyage en tenant compte des perturbations. Dans [Sghaier, 2011], on étudie l'optimisation par modélisation graphique d'un système de covoiturage dynamique à des fins environnementales et économiques. L'optimisation d'un système d'information distribué est un point commun entre tous ces travaux indiqués et les nôtres.

Dans [Satunin et Babkin, 2014], on propose la modélisation d'un système de transport intelligent pour l'affectation des utilisateurs à des véhicules partagés. Le système repose sur la combinaison d'un système multi-agent avec une approche de vote combinatoire. L'objectif de ce travail est d'optimiser les affectations pour mieux satisfaire les utilisateurs. Dans [Meng et al., 2007], un système d'agents est combiné à un algorithme génétique afin de résoudre la problématique de génération automatiques des examens pour des élèves. Dans [Cardon et al., 2000], on propose un système d'agents pour résoudre un problème de planification de type job-shop. Une approche génétique est adoptée en combinaison avec un protocole de négociation pour la génération des nouveaux ordonnancements. L'objectif est de créer un système collaboratif qui permet d'améliorer un diagramme de Gantt.

Dans cette thèse, l'objectif principal est la conception d'un système d'information distribué innovant qui offre non seulement des services de transport mais aussi des services dans des domaines voisins afin de donner une expérience riche de mobilité ubiquitaire avancée. Nous proposons de combiner l'approche de modélisation orientée agent avec une approche de résolution basée sur les algorithmes génétiques.

## II.5 Comparatif des Systèmes Distribués et Positionnement

Dans un espace qualifié d'ubiquitaire, plusieurs ressources et moyens technologiques sont disponibles et peuvent être mis en œuvre en faveur d'une mobilité urbaine plus adaptée aux besoins économiques et environnementaux. Les objets intégrés dans un tel espace sont capables de communiquer les uns avec les autres et avec les TM des utilisateurs, ce qui implique des contraintes considérables. Ainsi, un SI de recherche et de composition de services d'aide à la mobilité adapté pour un espace doit prendre en considération plusieurs verrous scientifiques et technologiques :

- *Charge élevée* : le nombre très élevé d'utilisateurs, des requêtes simultanées et des préférences utilisateurs, une multitude de services et de fournisseurs de services ;
- *Système étendu* : la topologie largement distribuée des entités composantes de cet espace ;
- *Environnement hétérogène* : l'hétérogénéité non seulement des ressources mais aussi des dispositifs de communication ;
- *Environnement dynamique* : les utilisateurs se connectent et se déconnectent d'une manière aléatoire, les informations sont dynamiques et changent continuellement.

Dans ce chapitre nous avons présenté une revue de la littérature sur la modélisation et le développement des systèmes d'information étendus. Dans le tableau II.1 nous proposons une étude comparative entre ces différents systèmes. Chaque système possède des points forts et des points faibles selon les critères

que nous nous sommes fixés. En effet, les critères dépendent de la problématique que nous traitons dans cette thèse.

Jusqu'à présent il n'y a pas de système d'aide au déplacement avancé adapté pour des espaces à grande échelle telle qu'une ville entière. La ville ubiquitaire augmentée d'une couche d'ordinateurs interconnectés peut être vue comme un environnement distribué, hautement sollicité et en communication continue avec les TM des utilisateurs qui peuvent être, au même temps, des demandeurs ou bien des fournisseurs d'IMA. Les architectures classiques centralisées ne peuvent pas répondre à notre objectif puisque la ville se caractérise par une multitude de services et un très grand nombre d'utilisateurs. Donc nous avons besoin d'une architecture avancée qui doit être robuste, flexible et extensible, capable de gérer le flux débordant de données et d'optimiser la recherche et la distribution des IMA. Une architecture distribuée, basée sur un SMA et inspirée des systèmes P2P est convenable pour ce type d'environnement. Elle a les avantages d'être flexible, étendue et susceptible d'intégrer différents algorithmes d'optimisation.

## II.6 Conclusion

Notre objectif dans ce travail est de concevoir et optimiser un système d'information distribué d'aide à la mobilité dans un espace ubiquitaire. Il existe plusieurs modèles de systèmes d'information distribués mais ne correspondent pas tous aux critères fixés dans ce travail.

En effet, l'étude bibliographique que nous avons mené, nous motive à adopter une approche qui associe les systèmes multi-agents à l'optimisation évolutionnaire distribuée et à s'inspirer de la logique des systèmes pair-à-pair. Dans le chapitre suivant, nous présentons l'architecture du système et nous proposons les approches et les méthodes développées. Nous exposons en particulier le comportement et l'évolution dynamiques du système pour le traitement des requêtes simultanées et l'optimisation de la recherche des services.

TABLE II.1: Comparaison des différents systèmes présentés dans l'état de l'art

| **Propriété** | | **Système P2P** | **Système de Nuage** | **Système de Grille** | **Système d'Agents** |
|---|---|---|---|---|---|
| Environnement | Dynamique | oui (les pairs se connectent et se déconnectent) | oui (les clients se connectent et se déconnectent) | oui (les clients se connectent et se déconnectent) | oui (les agents se connectent et se déconnectent) |
| | Hétérogène | oui (les clients sont multi-plateforme, multi-dispositif) | non (les clients ne sont pas multi-plateforme) | oui (les clients sont multi-plateforme) | oui (les clients sont multi-plateforme, multi-dispositif) |
| Modèlisation | Centralisée | non | oui | oui | oui |
| | Hybride | oui | non | non | oui |
| | Totalement distribuée | oui (en utilisant les réseaux virtuels ou superposés) | non (pas de comportement serveur chez les clients | non (pas de comportement serveur chez les clients | oui (les agents peuvent changer de rôle et chercher les services sans l'intervention d'une unité centrale) |
| Autonomie | | non (l'intervention de l'utilisateur est essentielle pour des tâches de gestion) | non (l'intervention de l'utilisateur est essentielle pour des tâches de gestion) | non (l'intervention de l'utilisateur est essentielle pour des tâches de gestion) | oui (l'intervention de l'utilisateur est facultative et dépend des tâches) |
| Mobilité du code | | non | non | non | oui (agent mobile) |
| Comportement | Coopération | oui (partage des capacités de stockage) | non | oui (partage des ressources de traitement et de stockage ) | oui (partage des ressources de traitement et de stockage) |
| | Coordination | oui (distribué ou attribué à une unité centrale) | oui (attribué à une unité centrale) | oui (attribué à une unité centrale) | oui (distribué ou attribué à une unité centrale) |
| | Négociation | oui (basique, pour la récupération d'un contenu) | non (pas d'interaction entre les clients) | non (pas d'interaction entre les clients | oui (avancée, basée sur des protocoles tels que CNET, PAAN, ANTS (chap. IV). Avec des stratégies tels que les ventes aux enchères) |
| | Compétition | non (les pairs sont indépendants) | non (les clients sont indépendants) | oui (les clients peuvent être menés à des calculs compétitifs | oui (les agents peuvent être compétitifs lors de la résolution d'un problème) |

# Chapitre III

# Proposition d'un Système d'Agents Optimisateurs pour la Recherche et la Distribution des Informations de Mobilité Avancée

**Sommaire**

## III.1 Introduction

Notre objectif est de concevoir et de développer une plateforme de recherche et de composition des services d'aide à la mobilité (PRoSAM). Néanmoins, dans le contexte d'un espace ubiquitaire, plusieurs défis se présentent : croissance exponentielle des services disponibles sur les réseaux distribués étendus, nombre élevé de demandes utilisateurs qu'il faut optimiser et satisfaire tout en respectant les contraintes sur le coût, le temps de réponse et la qualité de service.

En effet, une information demandée correspond à un service, qui peut être proposé différemment, par plusieurs fournisseurs d'information, en concurrence, avec différents coûts, temps de réponse et tailles de données. Sachant qu'un ensemble de services peut être sollicité simultanément par un nombre très important d'utilisateurs, des mesures de conception et d'optimisation spécifiques, que nous présentons dans ce chapitre, sont prises en considération afin de s'adapter à ce nouveau contexte.

La suite du chapitre est organisée en trois parties : la première partie présente la formulation du problème et évoque en particulier la modélisation des requêtes utilisateurs. La deuxième partie propose l'architecture du système sous forme d'entités autonomes en interaction et présente le comportement de ces différentes entités (i.e. les agents). Finalement, la troisième partie présente les algorithmes et les méthodes de résolution et de génération des itinéraires des agents mobiles responsables de la collecte des informations de mobilité avancée (IMA).

## III.2 Formulation du Problème

Les services d'aide à la mobilité dans un espace ubiquitaire sont demandés par les utilisateurs à partir de leurs multiples dispositifs ubiquitaires, créant ainsi d'intenses flux d'information. Le système d'information d'aide à la mobilité doit dans ce cas gérer les flux des requêtes utilisateurs situés dans cet espace tout en optimisant le coût, délai de réponse, quantité de données, etc. Dans cette partie nous présentons une formalisation du problème et nous détaillons les différents variables proposés (regroupés dans la table III.2 à la fin de ce paragraphe).

### III.2.1 Préliminaires

Les requêtes utilisateurs sont nombreuses et formulées via différents dispositifs (Smartphones, tablettes, ordinateur portables, etc.). Nous considérons que toutes ces entités, créant une topologie distribuée, sont représentées par des nœuds de même poids dans le réseau. En effet, il n'y aura pas un nœud plus important qu'un autre et il n'y a pas de dépendance entre ces nœuds à l'exception de quelques informations de routage fournies par le système (détaillées dans le chapitre IV). Tous les nœuds sont autonomes et coopèrent afin de garantir la disponibilité des services pour l'ensemble des utilisateurs connectés au système. Les capacités de stockage et de traitement offertes par l'ensemble des nœuds permettent d'optimiser le comportement du système vis-à-vis de la stabilité, la disponibilité et l'élasticité. C'est à dire la capacité d'ajuster d'une façon automatique les ressources en termes d'informations d'aide à la mobilité au sein du système dans le but de maintenir ses performances en cas de forte demande.

Par conséquent, le système doit intégrer un module d'optimisation qui sera responsable de la minimisation du coût et du temps de réponse pour les requêtes utilisateurs. En effet le système propose un ensemble de services élémentaires qui peuvent être fournis par plusieurs fournisseurs de services du RDTC (Réseau Distribué de Transport Comodal) avec différents coûts, temps d'exécution et quantités de données transférées.

### III.2.2 Description du Problème

Le système proposé dans cette thèse [Bousselmi et al., 2014, Bousselmi et al., 2013] consiste en un Plateforme de Recherche et de composition des Services d'Aide à la Mobilité (PRoSAM). PRoSAM est définie par un ensemble de connecteurs et de coalitions de clients (une coalition par connecteur). Les connecteurs sont répartis sur plusieurs zones géographiques (les espaces ubiquitaires) et permettent la gestion et la supervision des coalitions (appelées aussi voisinages collaboratifs) afin de garantir un accès optimisé aux services du RDTC par les utilisateurs dans chaque zone. Ces utilisateurs sont représentés par des agents mobiles qui sont capables de changer leurs rôles d'une manière dynamique et de demander des IMA offertes originellement par les fournisseurs de services dans un RDTC.

La Fig. III.1 présente une schématisation du système (connecteurs + voisinages collaboratifs) en interaction avec le RDTC sous forme de niveaux. La description du problème consiste à définir les paramètres de chacun de ces niveaux. Le premier niveau porte sur les fournisseurs originaux des services d'aide à la mobilité. Le deuxième niveau se focalise sur le noyau d'optimisation, de recherche

et de distribution des services (i.e les connecteurs). Enfin, le troisième niveau s'intéresse aux voisinages collaboratifs (i.e. l'ensemble des utilisateurs liés à chaque connecteur).

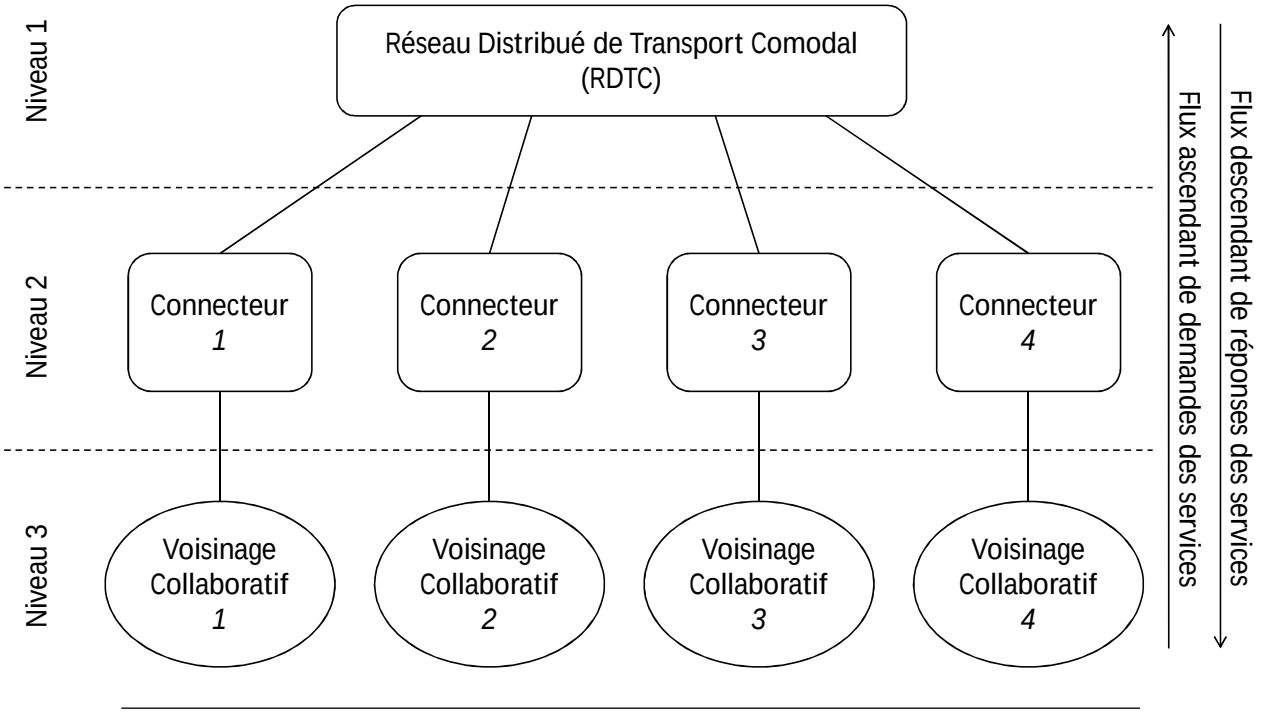



FIGURE III.1: Schéma de l'architecture globale du système.

#### III.2.2.1 Niveau 1 : RDTC

**Définition 1.** ***Service & Fournisseur de Service*** : un service est une fonctionnalité ou une partie d'une fonctionnalité fournie par un composant logiciel afin d'accomplir une tâche bien déterminée. Un service d'aide à la mobilité offre des informations ou une partie d'information en lien avec la mobilité de l'utilisateur. Un service peut être n'importe quelle ressource pouvant être partagée et utilisée dans un réseau. Dans ce cas, le fournisseur de service est l'entité responsable de le proposer, le mettre à jour et d'en assurer la disponibilité.

**Service** $(s_m)$ : un identifiant $ids_m$ est défini pour chaque service élémentaire $s_m$ du RDTC accessible par l'ensemble des connecteurs. L'ensemble de ces services est noté par $\mathbb{S} = \{ids_1, ids_2, .., ids_M\}$. Donc $\mathbb{S} = \{ids_m\}_{m\in[\![1,M]\!]}$, où $M$ représente le nombre total des services proposés par le RDTC.

**Fournisseur** $(F_v)$ : la récupération d'une réponse à un service depuis le RDTC nécessite la sélection d'un fournisseur de service $F_v$ par un connecteur. L'ensemble des fournisseurs de services est noté par $\mathbb{F} = \{F_1, F_2, .., F_V\}$. Donc $\mathbb{F} = \{F_v\}_{v\in[\![1,V]\!]}$, où $V$ représente le nombre total des fournisseurs de

services du RDTC. Nous supposons que :

$$\forall ids_m \in \mathbb{S}, \exists F_v \in \mathbb{F}/F_v \text{ fournit } ids_m \tag{III.1}$$

**Coût** ($c_{m,v}$) : chaque service $s_m$ proposé par un serveur $F_v$ possède un coût noté par $c_{m,v}$. Le coût normalisé [1] correspondant à $c_{m,v}$ est notée par $\bar{c}_{m,v}$.

**Quantité de données** ($q_{m,v}$) : le résultat de l'invocation d'un service $s_m$ proposé par un serveur $F_v$ correspond à une quantité de données notée par $q_{m,v}$. La quantité de données normalisée correspondant à $q_{m,v}$ est notée par $\bar{q}_{m,v}$.

**Temps d'exécution** ($te_{m,v}$) : le temps d'exécution est la durée nécessaire pour le traitement d'un service $s_m$ par un serveur $F_v$.

La table III.1 représente un exemple généralisé d'une table de services qui regroupe les services et les fournisseurs de ces services. Le symbole "–" signifie que le service n'est pas proposé par le fournisseur correspondant. Cette table est utilisée par les connecteurs afin d'identifier les fournisseurs originaux des services demandés par les utilisateurs du système.

TABLE III.1: Table de services à $M$ services et $V$ fournisseurs de services.

| | $F_1$ | $F_2$ | **...** | $F_{V-1}$ | $F_V$ |
|---|---|---|---|---|---|
| $s_1$ | $(c_{1,1}, te_{1,1}, q_{1,1})$ | $(c_{1,2}, te_{1,2}, q_{1,2})$ | ... | – | $(c_{1,V}, te_{1,V}, q_{1,V})$ |
| $s_2$ | – | $(c_{2,2}, te_{2,2}, q_{2,2})$ | ... | – | – |
| **...** | ... | ... | ... | ... | ... |
| $s_{M-1}$ | $(c_{1,M-1}, te_{1,M-1}, q_{1,M-1})$ | – | ... | – | – |
| $s_M$ | $(c_{M,1}, te_{M,1}, q_{M,1})$ | – | ... | $(c_{M,V-1}, te_{M,V-1}, q_{M,V-1})$ | $(c_{M,V}, te_{M,V}, q_{M,V})$ |

#### III.2.2.2 Niveau 2 : Connecteurs

La structure logicielle d'un connecteur est détaillée dans le paragraphe III.3.2.2.

**Connecteur** ($C_j$) : un connecteur $C_j$ est le composant du système responsable de l'optimisation, la recherche et la distribution des services d'aide à la mobilité depuis le RDTC vers le voisinage collaboratif. Un connecteur est composé d'un ensemble d'agents collaborateurs (III.3.2.2). L'ensemble des connecteurs est défini par $\mathbb{C} = \{C_1, C_2, .., C_L\}$. Donc $\mathbb{C} = \{C_j\}_{j \in [\![1,L]\!]}$. $L$ représente le cardinal de $\mathbb{C}$.

**Société d'agents d'un connecteur** ($X_j^t$) : une société d'agents $X_j^t$ créée à l'instant $t$ au sein d'un connecteur $C_j$, est l'entité responsable du traitement des requêtes utilisateurs parvenues à ce

1. L'algorithme d'évaluation des solutions effectue l'agrégation du coût et de la quantité de données d'un service donné. Pour cela, nous utiliserons des coûts et des quantités de données normalisés.

connecteur à l'instant $t$. Une société d'agents ne peut traiter qu'un nombre fini de requêtes utilisateurs simultanées. Ce nombre est appelé seuil de traitement et noté par $\gamma_j$. Si à un instant $t'$, le nombre des requêtes parvenues au connecteur $C_j$ dépasse $\gamma_j$, ceci déclenche la création d'une nouvelle société d'agents $X_j^{t'}$ pour assister les sociétés existantes. Les comportements des agents optimisateurs qui composent chaque société d'agents sont présentés dans le paragraphe III.4.

**Définition 2.** ***Période d'inactivité*** : cette période est définie comme étant la durée de temps au bout de laquelle se déclenche l'état d'inactivité d'une société d'agents dans un connecteur. Cette période est notée $Tin$.

**Définition 3.** ***Période d'acquisition*** : il s'agit de l'intervalle de temps pendant lequel une société d'agents collecte des requêtes utilisateurs. Cette fenêtre de temps étant très petite, les requêtes reçues pendant cette période sont considérées comme "simultanées". Cette période est noté par $Tacq$. Il s'agit d'une constante système.

**Services Demandés** ($\mathbb{S}_j^t$) : l'ensemble des services demandés par les utilisateurs d'un connecteur $j$ à un instant $t$ est défini par $\mathbb{S}_j^t$ et son cardinal est noté par $M_j^t$, avec $\mathbb{S}_j^t \subseteq \mathbb{S}$.

**Fournisseurs Sélectionnés** ($\mathbb{F}_j^t$) : pour tout ensemble de services demandés $\mathbb{S}_j^t$ à un instant $t$ sur un connecteur $j$, un ensemble de fournisseurs de services est sélectionné pour répondre à ces demandes. Cet ensemble est défini par $\mathbb{F}_j^t$ et son cardinal est noté par $V_j^t$, avec $\mathbb{F}_j^t \subseteq \mathbb{F}$.

#### III.2.2.3 Niveau 3 : Voisinages Collaboratifs

La structure logicielle d'un voisinage collaboratif est détaillée dans le paragraphe III.3.2.1.

**Voisinage Collaboratif** ($K_j^t$) : un voisinage collaboratif $K_j^t$, appelé aussi coalition, est un composant virtuel du système défini par l'ensemble des clients liés à un connecteur $C_j$ à l'instant $t$. L'ensemble des coalitions à l'instant $t$ est noté par $\mathbb{K}(t) = \{K_1^t, K_2^t, .., K_L^t\} = \{K_j^t\}_{j \in [\![1,L]\!]}$.

Nous nous positionnons dans le cas où les clients du transport se connectent au système avec des dispositifs mobiles. Ces derniers sont représentés par des agents mobiles que nous appelons *agents utilisateurs* (III.4.1).

**Définition 4. Rôle *Client Passif*** : le rôle *client passif* est le rôle attribué à chaque nouveau client (i.e. agent utilisateur) qui vient de rejoindre le voisinage collaboratif. Une fois attribué à un utilisateur, ce rôle ne lui permet que de demander des services.

**Client Passif** ($a_{h,j}$) : un membre d'une coalition $K_j^t$ est noté par $a_{h,j}$ quand un rôle client passif lui est attribué. L'ensemble des clients passifs liés à un connecteur $C_j$ à l'instant $t$ est défini par $A_j^t = \{a_{1,j}, a_{2,j}, .., a_{N_j^t,j}\} = \{a_{h,j}\}_{h\in[\![1,N_j^t]\!]}$. $N_j^t$ représente le nombre total de clients passifs connectés à $C_j$ à l'instant $t$.

**Définition 5. Rôle *Client Actif*** : le rôle *client actif* est le rôle attribué à un membre du voisinage collaboratif pour lui permettre non seulement de demander des services mais aussi de fournir à son tour les services qu'il a déjà demandé et reçu.

**Client Actif** ($\hat{a}_{h,j}$) : un membre d'une coalition $K_j^t$ est noté par $\hat{a}_{h,j}$ quand un rôle client actif lui est attribué. L'ensemble des clients actifs liés à un connecteur $C_j$ à l'instant $t$ est défini par $\hat{A}_j^t = \{\hat{a}_{1,j}, \hat{a}_{2,j}, .., \hat{a}_{\hat{N}_j^t,j}\} = \{\hat{a}_{h,j}\}_{h\in[\![1,\hat{N}_j^t]\!]}$. $\hat{N}_j^t$ représente le nombre total des clients actifs connectés à $C_j$ à l'instant $t$. En outre, $\hat{A}_j^t \bigcap A_j^t = \varnothing$ et $\hat{A}_j^t \bigcup A_j^t = K_j^t$.

Le comportement d'un agent utilisateur, pouvant avoir le rôle d'un client actif ou passif, est détaillé dans le paragraphe III.4.1.

**Requête Client** ($req_{h,j}$) : chaque client $a_{h,j}$ ou $\hat{a}_{h,j}$ peut envoyer une requête $req_{h,j}$ à un récepteur appartenant à l'ensemble $C_j \cup \hat{A}_j^t \backslash \{\hat{a}_{h,j}\}$. Une requête est définie par $req_{h,j} = \{ids_1, ids_2, .., ids_{W_{h,j}}\}$. Donc $req_{h,j} = \{ids_w\}_{w\in[\![1,W_{h,j}]\!]}$. $W_{h,j}$ représente le nombre total des services demandés par le client $h$ lié au connecteur $j$.

$$\{ \bigcup_{j\in[\![1,L]\!]} \bigcup_{h\in[\![1,N_j^t+\hat{N}_j^t]\!]} req_{h,j}\} \subseteq \mathbb{S} \tag{III.2}$$

**Date de réponse au plus tard** ($dmax_{h,j}$) : chaque requête $req_{h,j}$ d'un client $h$ appartenant à un voisinage collaboratif $K_j^t$ possède une date de réponse au plus tard. Cette date est notée par $dmax_{h,j}$.

**Requêtes Simultanées** ($R_j^t$) : l'ensemble des requêtes simultanées parvenues au connecteur $C_j$ à partir de l'instant $t$ et pendant la période d'acquisition $Tacq$ est noté par $R_j^t$.

$$R_j^t = \bigcup_{h\in[\![1,N_j^t+\hat{N}_j^t]\!]} req_{h,j} \tag{III.3}$$

L'exemple de la Fig.III.2 illustre une requête d'un client $a_{3,1}$ (le client numéro 3 rattaché au connecteur numéro 1) demandant un ensemble de services : {$ids_1$, $ids_3$, $ids_4$, $ids_{10}$, $ids_{12}$, $ids_{25}$, $ids_{17}$}. Les fournisseurs potentiels de chaque service sont identifiés. A titre d'exemple, le service $s_1$ peut être proposé par deux clients actifs $\hat{a}_{1,1}$ et $\hat{a}_{10,1}$ ; par contre $s_3$ n'est proposé que par $C_1$.

Du côté connecteur, le même service peut être proposé par plusieurs fournisseurs de services à différents coûts, délais de traitement, quantité de données, etc. La sélection des fournisseurs potentiels est

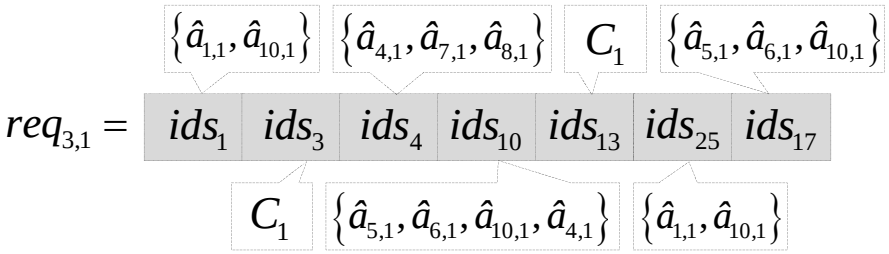



FIGURE III.2: Exemple d'identification des fournisseurs de services.

gérée par les critères d'optimisation utilisés par chaque connecteur. Du côté client, les services sont récupérés auprès du connecteur ; ainsi la réponse reçue est optimale. Ces services peuvent être dans ce cas "achetés" ou "vendus" par les membres de la coalition pendant une période de temps bien déterminée. L'échange des services au sein d'un voisinage collaboratif fera l'objet du chapitre IV.

TABLE III.2: Récapitulatif des variables principales du système

| **Notations** | **Descriptions** |
|---|---|
| $a_{h,j}$ | client passif numéro $h$ lié à un connecteur $j$ |
| $\hat{a}_{h,j}$ | client actif numéro $h$ lié à un connecteur $j$ |
| $\mathbb{C}$ | ensemble des connecteurs disponibles |
| $C_j$ | connecteur numéro $j$ |
| $c_{m,v}$ | coût d'un service $s_m$ proposé par un serveur $F_v$ |
| $\bar{c}_{m,v}$ | coût normalisé d'un service $s_m$ proposé par un serveur $F_v$ |
| $dmax_{h,j}$ | date de réponse au plus tard d'une requête $req_{h,j}$ |
| $\mathbb{F}$ | ensemble des fournisseurs de services appartenant au RDTC |
| $\mathbb{F}_j^t$ | ensemble des fournisseurs de services sélectionnés pour satisfaire les demandes envoyées à l'instant $t$ par les utilisateurs liés à un connecteur $j$ |
| $K_j^t$ | voisinage collaboratif rattaché à un connecteur $j$ à l'instant $t$ |
| $L$ | cardinal de l'ensemble $\mathbb{C}$ |
| $M$ | cardinal de l'ensemble $\mathbb{S}$ |
| $M_j^t$ | cardinal de l'ensemble $\mathbb{S}_j^t$ |
| $N_j^t$ | nombre de clients passifs dans un voisinage collaboratif $K_j^t$ à l'instant $t$ |
| $\hat{N}_j^t$ | nombre de clients actifs dans un voisinage collaboratif $K_j^t$ à l'instant $t$ |
| $q_{m,v}$ | quantité de données d'un service $s_m$ proposé par un serveur $F_v$ |
| $\bar{q}_{m,v}$ | quantité de données normalisée d'un service $s_m$ proposé par un serveur $F_v$ |
| $req_{h,j}$ | requête d'un client $h$ liée à un connecteur $j$ |
| $R_j^t$ | ensemble de requêtes simultanées reçues par un connecteur $j$ à l'instant $t$ |
| $\mathbb{S}$ | ensemble des services proposés par le RDTC |
| $\mathbb{S}_j^t$ | ensemble des services demandés à un instant $t$ par les utilisateur d'un connecteur $j$ |
| $Tacq$ | période d'acquisition d'un ensemble $R_j^t$ |
| $te_{m,v}$ | temps de traitement d'un service $s_m$ proposé par un serveur $F_v$ |
| $Tin$ | période d'inactivité de chaque connecteur |
| $V$ | cardinal de l'ensemble $\mathbb{F}$ |
| $V_j^t$ | cardinal de l'ensemble $\mathbb{F}_j^t$ |
| $X_j^t$ | société d'agents créée à l'instant $t$ au sein d'un connecteur $j$ |

Les utilisateurs du système sont formulent leurs requêtes en utilisant leurs terminaux mobiles (TM).

Une requête utilisateur correspond à un ensemble de services pouvant être proposés par un ensemble de :

– fournisseurs de services originaux distribués au sein du RDTC ;
– clients actifs proposant des services qu'ils ont déjà demandé et reçu.

Afin de prendre en considération le nombre élevé de services pouvant être demandés par les clients, nous proposons de composer la requête en se basant sur la technique de hachage. En effet lorsqu'un service est demandé, un identifiant correspondant à ce service est affecté à la requête. Cet identifiant est le résultat du hachage (appelé aussi condensé) du nom du service par exemple. Cette technique permettra une identification optimisée des services demandés en réduisant le temps de recherche considérablement. En effet, la recherche linéaire classique est $O(n)$ (avec $n$ le nombre d'éléments dans la liste de recherche), ce qui n'est pas le cas avec les tables de hachages. Ces dernières permettent une complexité constante $O(1)$. D'où l'intérêt de l'adoption de cette technique de stockage des données. Il faut aussi préciser que ce travail ne traite pas les problèmes de conception des fonctions de hachage et de collision des condensés.

#### III.2.2.4 Exemple de Composition des Requêtes Utilisateurs

Pour illustrer la composition des requêtes nous considérons un exemple simplifié composé de :

– 1 connecteur $C_1$ qui permet l'accès aux services suivants : $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9\}$ ;
– 4 nouveaux utilisateurs $a_{5,1}$, $a_{6,1}$, $a_{7,1}$ et $a_{8,1}$ qui rejoignent le voisinage collaboratif ;
– ces utilisateurs composent leurs requêtes simultanément et demandent respectivement les ensembles de services : $\{s_2, s_3, s_5, s_7\}$, $\{s_1, s_3, s_5, s_8, s_9\}$, $\{s_4, s_5, s_6, s_7, s_8\}$ et $\{s_4, s_5, s_6\}$.

Si l'utilisateur $a_{6,1}$ demande les services $s_1$, $s_3$, $s_5$, $s_8$ et $s_9$, alors sa requête sera comme suit :

| | | | | |
|---|---|---|---|---|
| $ids_1$ | $ids_3$ | $ids_5$ | $ids_8$ | $ids_9$ |

où $ids_i$ est la valeur de hachage correspondant au service $s_i$, $i \in \{1, 3, 5, 8, 9\}$. Dans le reste du rapport, cette valeur est appelée identifiant du service.

De plus, nous considérons que le système proposé doit être en mesure de gérer les requêtes utilisateurs simultanées. En reprenant l'exemple ci-dessus, une première étape de gestion de la simultanéité des requêtes consiste à identifier les similarités entre ces requêtes comme le présente le tableau III.3. A titre d'exemple, le service $s_5$ et demandé quatre fois par les requêtes $req_{5,1}$ et $req_{6,1}$, $req_{7,1}$ et $req_{8,1}$. Par conséquent, au lieu d'effectuer quatre fois la recherche du service $s_5$, le système se contente de chercher le service une seule fois pour les requêtes qui demandent ce même service quasi-simultanément.

TABLE III.3: Exemple illustrant la redondance des services demandés.

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $req_{5,1}$ | | × | × | | × | | × | | |
| $req_{6,1}$ | × | | × | | × | | | × | × |
| $req_{7,1}$ | | | | × | × | × | × | × | |
| $req_{8,1}$ | | | | × | × | × | | | |

En tenant compte de l'aspect de redondance des demandes de services, il devient intéressant de passer par une étape de décomposition des requêtes simultanées avant de commencer le processus de recherche et de composition des réponses finales. Cette étape d'identification des similarités entre les requêtes est détaillée dans le paragraphe suivant.

## III.3 Modélisation Multi-Agent du Système Proposé

Dans cette partie du chapitre, nous proposons une architecture multi-agent pour la résolution du problème décrit précédemment. Le système proposé, appelé PRoSAM, est composé de deux sous systèmes principaux : les connecteurs et les coalitions. Dans ce qui suit, nous présentons le système dans sa globalité avant de détailler chacune de ses parties.

### III.3.1 Vue d'Ensemble du Système

L'objectif global du système d'agents proposé dans ce travail est de satisfaire les requêtes des utilisateurs tout en optimisant :

- la recherche et la distribution des services ;
- le coût total des services demandés ainsi que le temps de réponse aux requêtes ;
- la qualité de service globale en termes de disponibilité, délai de réponse, quantité de données transférées, etc.

Le système peut être vu comme étant deux sous-systèmes coopératifs qui interagissent afin d'atteindre les objectifs fixés ci-dessus. Le premier sous-système (le connecteur) est responsable des tâches d'optimisation et de routage des requêtes utilisateurs. Le connecteur est responsable aussi de la gestion des inscriptions des fournisseurs de services (intégrer une fonction d'authentification des utilisateurs au sein du connecteur fait partie des perspectives de ce travail). Le deuxième sous-système (la coalition) est responsable de l'interaction des utilisateurs et offre une plateforme fiable de partages des services en mode P2P. Les coalitions visent à améliorer la robustesse du système en réduisant la charge sur les fournisseurs originaux des services (au sein des connecteurs).

Un espace ubiquitaire est un environnement dynamique, évolutif et repose essentiellement sur une couche de communication sans-fil ; ce qui est contraignant lorsqu'il s'agit d'une charge réseau élevée (i.e. en termes de nombre d'utilisateurs et de flux de données). Pour cette raison, l'architecture du système doit être à la fois robuste et flexible afin de s'adapter au mieux à ces conditions. Afin d'atteindre cet objectif, un ensemble de techniques et de méthodes provenant des SMA, des systèmes P2P et de l'optimisation évolutionnaire sont combinées et introduites dans notre système. En effet, nous avons constaté que :

– tous les des travaux antérieurs sur les SIAM [2] orientés agents considèrent l'utilisateur comme étant un simple client qui demande des services en mode client-serveur [Zgaya, 2007, Sghaier, 2011, Feki, 2010] ;
– et que l'intégration de quelques fonctions serveur chez le client dans les systèmes P2P a démontré une grande capacité à réduire la charge sur les fournisseurs de services et à supporter un nombre très élevé d'utilisateurs [Merabti et El Rhalibi, 2004, Han et al., 2014, Pouwelse et al., 2005, Stoica et al., 2001].

Le système proposé, permet aux agents utilisateurs d'être capables de demander et d'offrir des services les uns aux autres. Cet échange d'informations n'est pas aléatoire mais plutôt orchestré par des indicateurs de la charge du réseau et du besoin des clients. Ces indicateurs sont détaillés dans le chapitre IV. L'architecture du système proposé dans cette thèse comporte 5 types d'agents logiciels, illustrés dans la table III.4. Les comportements de ces agents sont détaillés dans le paragraphe III.4.

TABLE III.4: Nomenclatures des 5 types d'agents du système

| **Nomenclatures** | **Descriptions** |
|---|---|
| ***Au*** | **A**gent **u**tilisateur |
| ***Ad*** | **A**gent **d**écomposeur |
| ***Ao*** | **A**gent **o**ptimisateur |
| ***Ar*** | **A**gent **r**amasseur |
| ***Ac*** | **A**gent **c**omposeur |

Dans ce qui suit, nous présentons l'architecture multi-agent proposée. La Fig. III.3 présente une illustration de cette architecture, où chaque nœud mobile (i.e. le terminal mobile du client) est représenté par un agent utilisateur $Au$.

2. Système d'Information d'Aide à la Mobilité

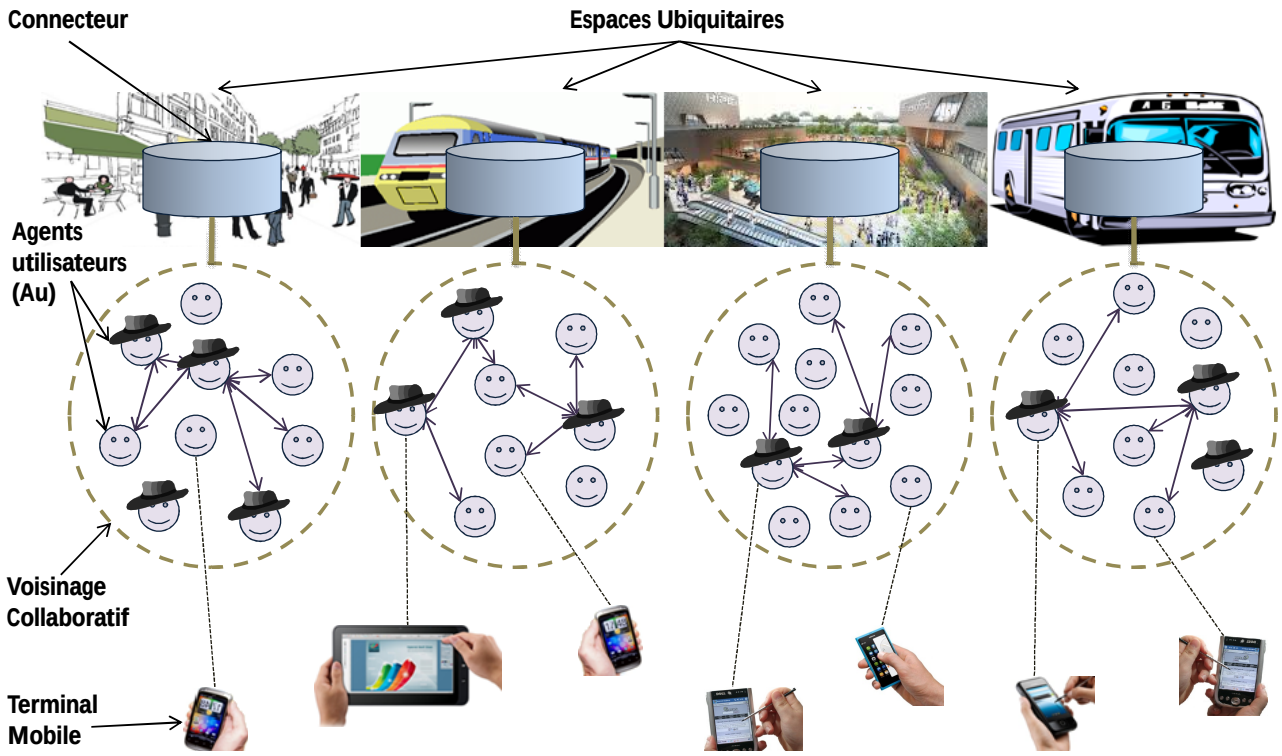



FIGURE III.3: Vue d'ensemble du système.

### III.3.2 Architecture Multi-Agent

Comme nous l'avons déjà mentionné, afin de répondre à la problématique présentée précédemment, nous proposons une architecture multi-agents composée de deux sous-systèmes principaux : les voisinages collaboratifs et les connecteurs. Dans cette partie, nous détaillons en premier lieu la structure des voisinages collaboratifs, en particulier la stratégie de changement de rôle dynamique. Ensuite, nous présentons la structure des connecteurs.

#### III.3.2.1 Structure d'un Voisinage Collaboratif (Coalition)

L'idée d'intégrer les dispositifs des utilisateurs dans les opérations de recherche et de découverte des services est inspirée des systèmes P2P que nous avons présentés dans le chapitre II. En effet, l'échange de services en mode pair-à-pair a pour objectif la réduction de la charge sur le serveur d'origine du service. Ceci permettra à notre système de supporter un nombre élevé d'utilisateurs. En d'autres termes, le système doit être en mesure de répondre à un très grand nombre de requêtes simultanées de recherche de contenu et de mise à jour.

Le voisinage collaboratif, ou coalition, représente la première pierre angulaire du système. Cette partie du système est conçue afin de renforcer sa flexibilité et sa robustesse. Les membres d'une coalition sont

les agents utilisateurs $Au$ qui sont capables de changer d'une manière dynamique leurs rôles (passif ou actif). En effet, ils sont capables d'acquérir le rôle d'un fournisseur de services et d'être en même temps des clients demandeurs d'autres services. Les rôles possibles qu'un agent utilisateur peut avoir dans le système : *client actif* ou *client passif*.

Le concept de changement de rôle pour un utilisateur est l'une des originalités de ce travail de thèse car les systèmes d'information conventionnels considèrent les utilisateurs comme de simples clients sans considérer la possibilité d'intégrer quelques fonctionnalités serveurs dans les applications front-end [3]. Dans un PRoSAM, les membres d'un voisinage collaboratif sont tous capables d'avoir non seulement le rôle d'un client (i.e. composition des requêtes, demande des services, affichage des résultats, etc.) mais aussi le rôle d'un serveur en hébergeant quelques informations au profit du système en général et de la coalition en particulier. Une stratégie de changement de rôle dynamique est fixée par le connecteur et basée sur un indice de redondance des services. Cette stratégie sera détaillée dans le chapitre IV.

#### III.3.2.2 Structure d'un Connecteur

L'architecture du connecteur est composée de quatre types d'agents logiciels : les Agents décomposeurs ($Ad$), les Agents optimisateurs ($Ao$), les Agents ramasseurs ($Ar$) et les Agents composeurs ($Ac$), comme le montre la Fig. III.4. Ces agents coopèrent et coordonnent leurs actions afin de traiter les requêtes utilisateurs simultanées depuis leur réception jusqu'à la génération des résultats finaux.

Ces agents sont responsables du processus de recherche et de composition des services d'aide à la mobilité dans la PRoSAM à partir des fournisseurs originaux du RDTC. Chaque agent est responsable d'un ensemble de tâches bien déterminé. Ils collaborent afin de proposer un service global de recherche, d'identification et de composition des IMA (Information de Mobilité Avancée) dans un espace ubiquitaire. En effet, les agents $Ad$ décomposent les requêtes et identifient les fournisseurs d'information potentiels. Ensuite, les agents $Ao$ s'occupent des ordonnancements services/fournisseurs et fournisseurs/agents $Ar$. Ensuite, les agents $Ar$ sont responsables de la recherche des IMA dans le RDTC. Enfin, les agents $Ac$ composent les réponses finales et les envoient aux utilisateurs correspondants. Nous présentons en détail le comportement de ces agents dans le paragraphe suivant.

Le fonctionnement du connecteur dépend principalement du flux des demandes reçues. En effet, si pendant une période d'inactivité $Tin$ aucune requête utilisateur n'est reçue, alors aucun agent n'existe dans le système. De cette manière, une grande capacité de calcul du connecteur peut être économisée

3. Dans le domaine de développement des applications et des logiciels, en particulier les systèmes d'information, l'interface à travers laquelle l'utilisateur interagit avec le système est désignée par le terme *front-end*. Quant à la partie du système qui s'occupe du traitement et de stockage des données, elle est désignée souvent par le terme *back-end*.

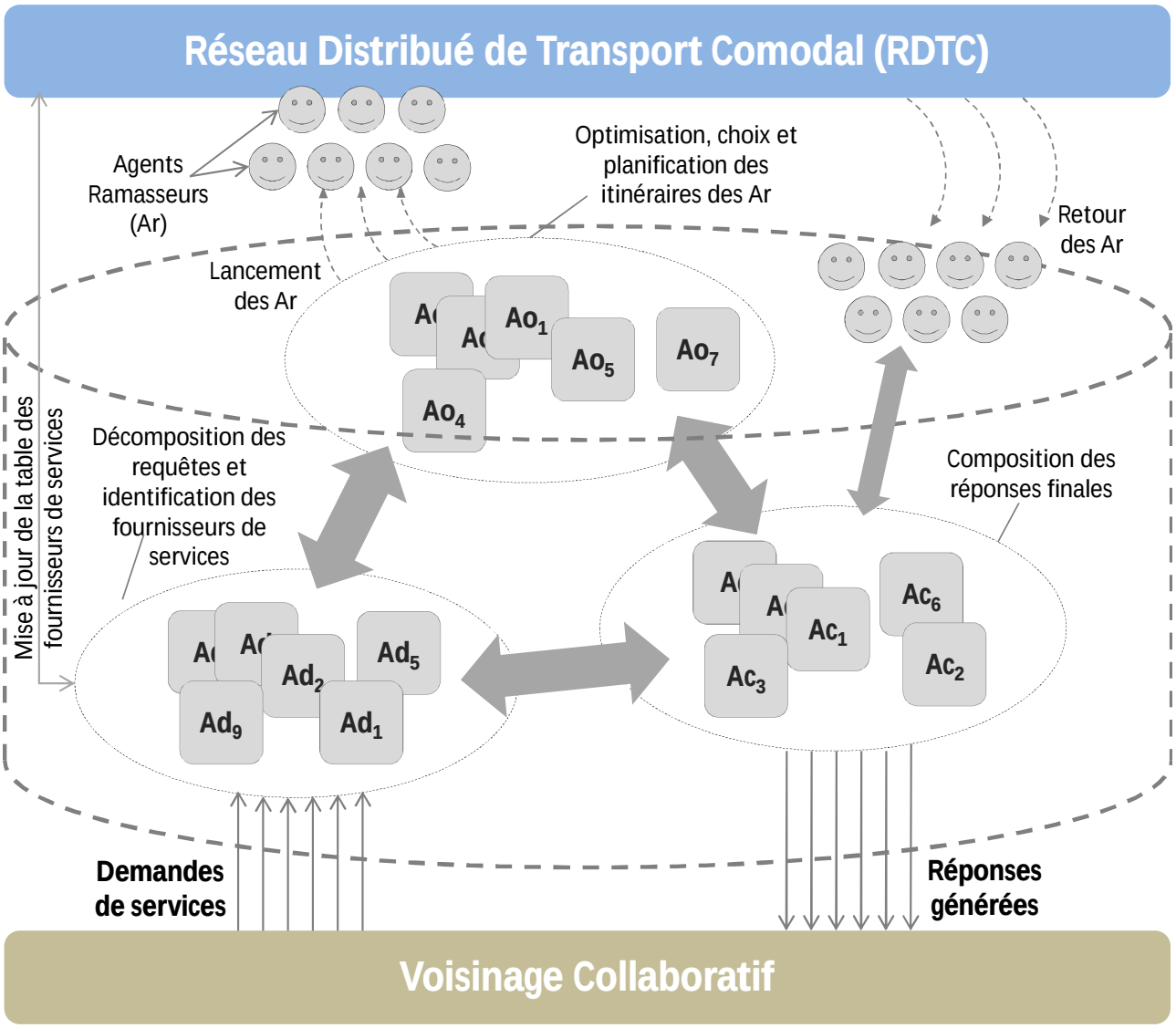



FIGURE III.4: Vue microscopique d'un connecteur.

et ces périodes d'inactivité peuvent être utilisées pour effectuer des mises à jour des bases de données (§III.5.3) du connecteur.

L'état du connecteur est défini donc principalement par l'évolution du voisinage collaboratif correspondant. En effet, plus le nombre de services demandés par les utilisateurs est élevé, plus le flux de données en circulation et la quantité de données traitée au sein du connecteur sont importants. Dans le but de minimiser les délais de réponses et faire face au même temps à la montée en charge, le voisinage collaboratif offre un ensemble de techniques, en particulier l'administration du changement dynamique des rôles des clients et le protocole de négociation adapté (présentés dans le chapitre IV) afin de supporter le connecteur et offrir une plateforme globale capable de gérer efficacement la recherche et la distribution des services d'aides à la mobilité.

## III.4 Spécification du Connecteur

Dans cette partie, nous détaillons la vue microscopique du connecteur présenté précédemment. Ainsi nous présentons les comportements des agents logiciels au sein du connecteur qui sont responsable de la recherche, la collecte et la distribution des services dans un voisinage collaboratif. Cette partie du système (i.e. le connecteur) comprend 4 types d'agents : un agent décomposeur $Ad$, un agent optimisateur $Ao$, un agent ramasseur $Ar$ et un agent composeur $Ac$ qui forment ce que nous avons appelé société d'agents au sein du connecteur. La deuxième partie du système (i.e. le voisinage collaboratif) qui se base principalement sur les agents utilisateurs $Au$ est détaillée dans le chapitre IV de ce rapport.

### III.4.1 Comportements des Agents du Connecteur

Dans la suite, nous détaillons le cycle de vie des agents $Ad$, $Ao$, $Ar$ et $Ac$ et nous exposons les diagrammes d'activité correspondants.

**Agent décomposeur (Ad)**. La Fig. III.5 présente le diagramme d'activité d'un agent décomposeur. Cet agent est responsable de la décomposition des requêtes utilisateurs reçues pendant la période d'acquisition $Tacq$. A l'issue de cette période de temps, l'agent $Ad$ vérifie si parmi l'ensemble des requêtes reçues il existe des requêtes qui peuvent être répondues directement depuis l'ETD. Si oui, il demande à l'agent $Ac$ répondre à ces requêtes. Ensuite il met à jour la base de données des requêtes reçues. Cette base de données est utilisée dans une prochaine étape par l'agent $Ac$ dans le processus de composition des réponses finales.

Pendant le processus de décomposition, chaque requête (non traitées) est dissociée en un ensemble de services élémentaires. Ce processus génère alors un ensemble d'identifiants de services non similaires (certains services peuvent être demandés par plusieurs utilisateurs) qui vont être récupérés depuis le Réseau Distribué de Transport Comodal dans une prochaine étape par les agents $Ar$. Les fournisseurs de services potentiels sont donc identifiés dans la table de services comme expliqué précédemment (§III.2).

Le cycle de vie de l'agent $Ad$ se termine après avoir envoyé la table de services à l'agent $Ao$. Ce cycle se répète pendant chaque période $Tacq$. Cet agent s'auto-détruit lorsqu'il atteint la durée d'inactivité $Tin$.

**Agent optimisateur (Ao)**. La Fig. III.6 présente le diagramme d'activité d'un agent optimisateur. Les serveurs d'IMA dans un RDTC peuvent proposer plusieurs services d'aide à la mobilité à différents coûts, délais de réponses et quantités de données. Après avoir reçu la table de services

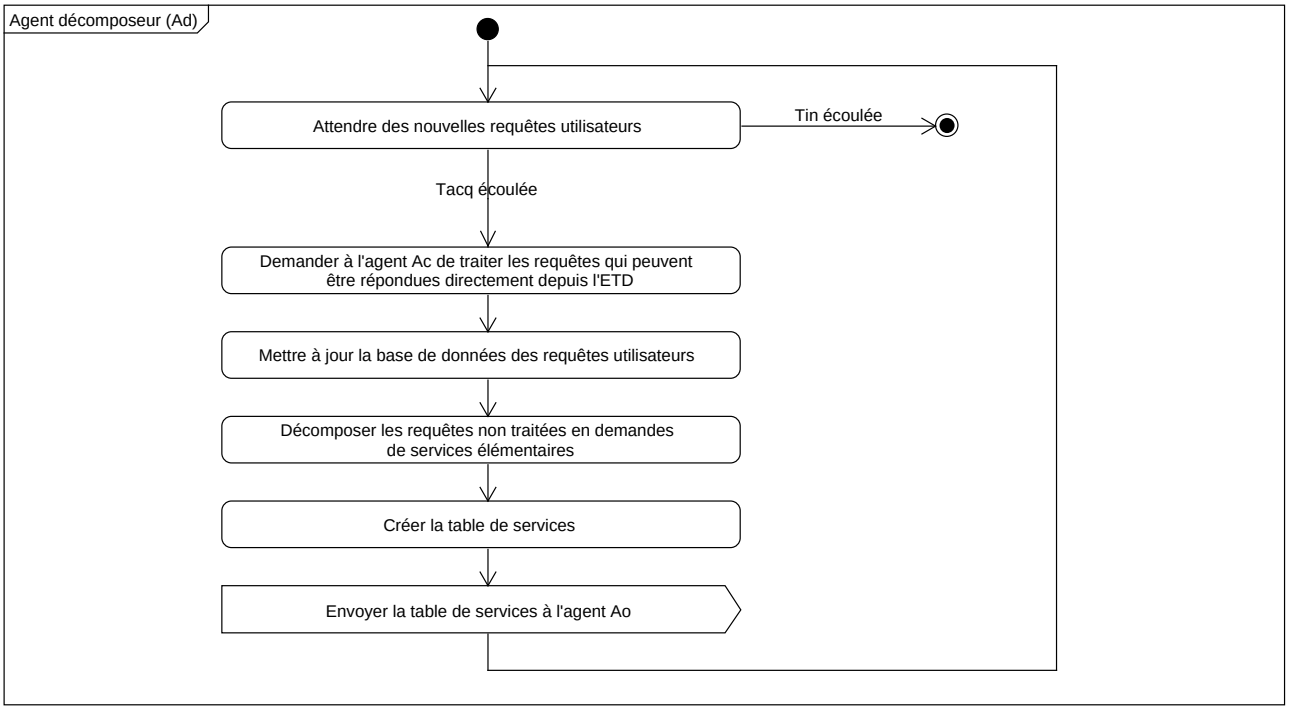



FIGURE III.5: Diagramme d'activité d'un Agent décomposeur (*Ad*).

envoyée par l'agent *Ad* et en utilisant une approche évolutionnaire (§III.5), l'agent *Ao* optimise les parcours (ensemble d'informations de routage) des agents *Ar* dans le RDTC.

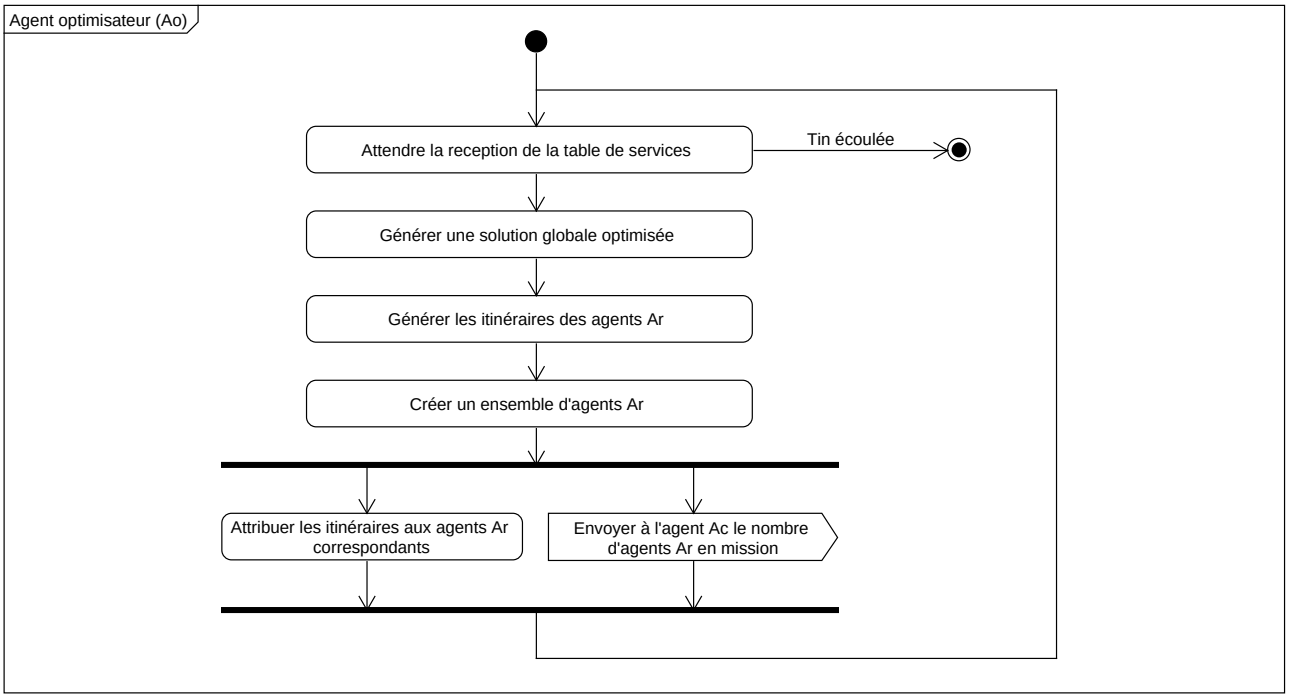



FIGURE III.6: Diagramme d'activité d'un Agent optimisateur (*Ao*).

En effet, l'agent *Ao* sélectionne la liste optimisée des fournisseurs de services pour la collecte des

données qui minimise le coût total ainsi que le délai de réponse au plus tard des services demandés. La tâche de l'agent optimisateur se termine lorsque ce dernier envoie les itinéraires aux agents $Ar$ correspondants qu'il a créé. Cet agent s'auto-détruit lorsqu'il atteint la durée d'inactivité $Tin$.

**Agent ramasseur (Ar)**. Le diagramme d'activité d'un agent ramasseur est illustré dans la Fig. III.7. L'agent $Ar$ est un agent logiciel mobile capable de se déplacer d'un nœud à l'autre à travers un réseau pour accomplir ses tâches. Dans notre système, nous utilisons ce paradigme afin de collecter les IMA à partir des nœuds du RDTC. L'agent $Ar$ se déplace selon l'itinéraire optimal généré par l'agent $Ao$, effectue ses tâches et revient en fin de parcours à son nœud d'origine : le connecteur à qui il appartient.

La taille de l'information qu'un agent $Ar$ doit collecter ne doit pas dépasser un seuil de capacité afin d'éviter la surcharge de données. L'agent $Ao$ doit donc prendre en considération cet aspect lors de la génération de la solution optimisée pour déduire les plans de routes (itinéraires) de ces agents mobiles. Avant la fin de son activité, l'agent $Ar$ revient au nœud de départ, communique les résultats collectés à l'agent $Ac$ et met à jour l'Entrepôt Temporaire de Données (ETD) selon l'indice de chaque service (§III.5.3). L'ETD est utilisé par le connecteur afin de réduire le nombre de recherches des services couramment demandés (§III.5.3).

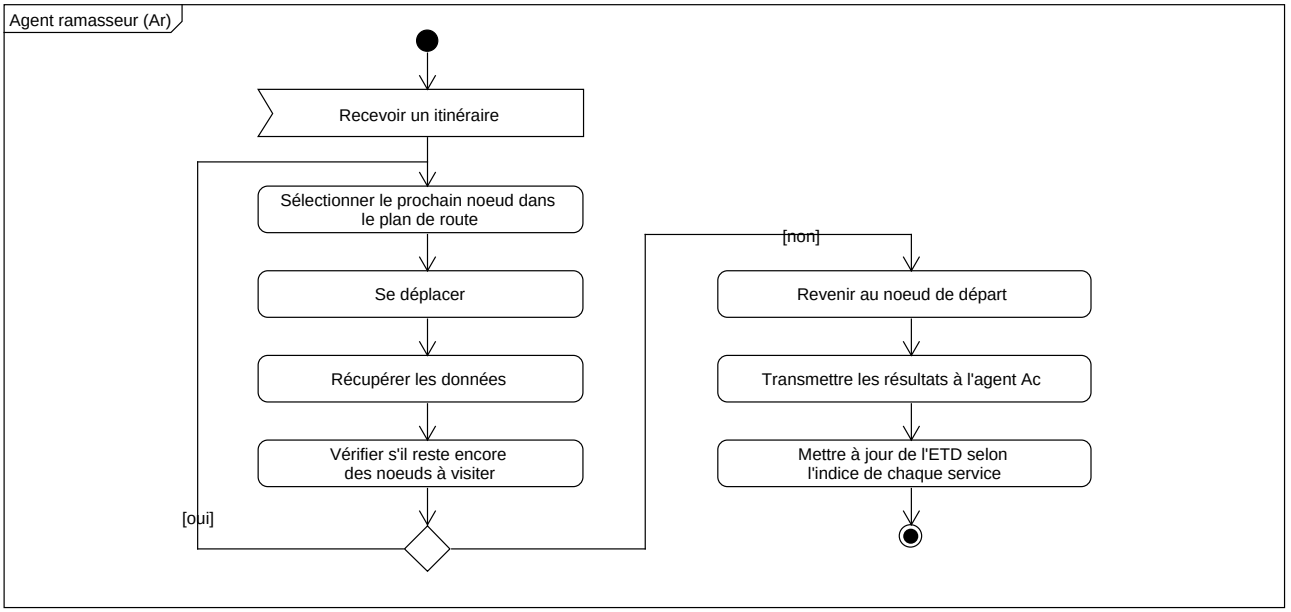



FIGURE III.7: Diagramme d'activité d'un Agent ramasseur ($Ar$).

**Agent composeur (Ac)**. Le comportement d'un agent composeur est illustré par le diagramme d'activité de la Fig. III.8. Cet agent se charge de la composition des réponses et la génération du résultat final aux utilisateurs ayant envoyé leurs demandes pendant la période $Tacq$. En utilisant la décomposition des requêtes reçues en tâches élémentaires et les données collectées par les agents $Ar$,

l'agent $Ac$ reconstitue les réponses finales pour les envoyer ensuite aux utilisateurs correspondants. Le processus de composition des réponses évolue selon la disponibilité des données collectées.

En effet, une requête ne peut être satisfaite que lorsque tous les services demandés ont été transmis à l'agent $Ac$. Dès qu'une réponse est prête, elle est immédiatement envoyée à l'utilisateur correspondant même si un autre utilisateur ayant composé sa requête avant n'a pas encore reçu sa réponse.

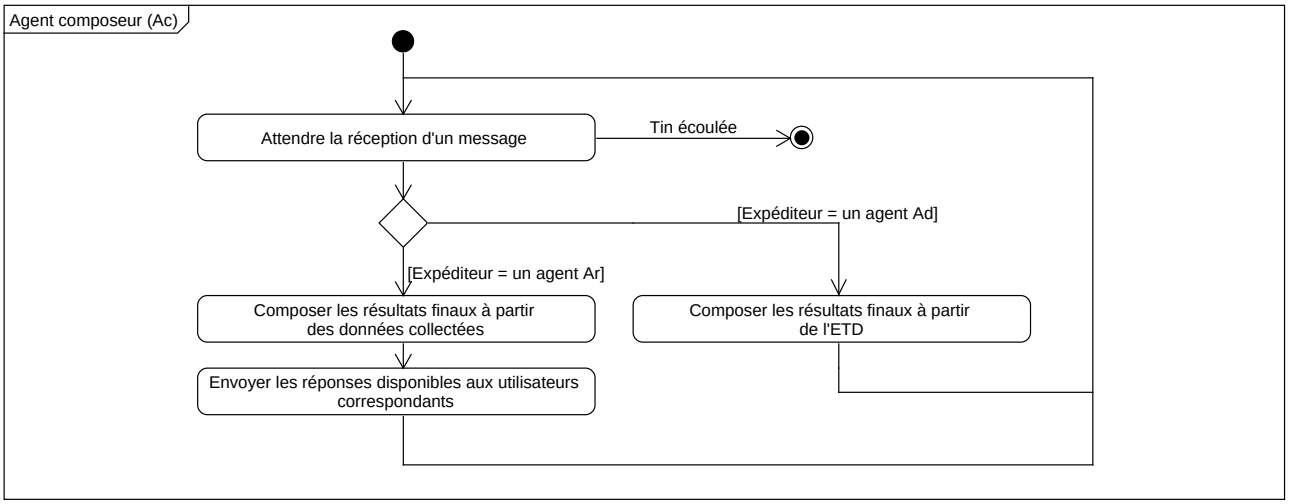



FIGURE III.8: Diagramme d'activité d'un Agent composeur ($Ac$).

### III.4.2 Comportement dynamique d'une Société d'Agents

Au départ, à un instant $t$, un connecteur $C_j$ est initialisé mais aucun agent (i.e. $Ad$, $Ao$, $Ar$ et $Ac$) n'est encore créé. Lorsque le premier agent utilisateur $Au$ se connecte au système et formule sa requête, la création d'une société d'agents $X_j^t$ se déclenche. Les requêtes simultanées reçues pendant une période $Tacq$ sont traitées par la même société d'agents $X_j^t$. Cet agent utilisateurs peut demander des services directement à son voisinage, dans ce cas aucune société d'agents n'est créée au sein du connecteur.

Si à l'instant $t + Tacq$, toutes les sociétés d'agents préalablement créées sont indisponibles, c'est à dire que ces sociétés sont entrain de traiter des requêtes reçues, alors la formulation de nouvelles requêtes par les agents utilisateurs déclenche la création d'une nouvelle société $X_j^{t+Tacq}$ et ainsi de suite. Si une société créée antérieurement redevient disponible, alors elle recommence à traiter les nouvelles demandes reçues. Ce cycle se répète infiniment et la condition d'arrêt est fixée par la période d'inactivité $Tin$. En effet, dès que la disponibilité d'une société atteint la durée $Tin$ alors elle est automatiquement détruite afin de libérer les ressources du système.

**Exemple d'un Scénario**. La Fig. III.9 représente un exemple d'un diagramme de temps du système. Dans ce scénario nous considérons un connecteur $C_1$ à $t = 0$ avec :

– une période d'acquisition $Tacq = 3s$ ;
– une période d'inactivité $Tin = 10s$ ;
– trois agents utilisateurs $a_{1,1}$, $a_{2,1}$, $a_{3,1}$ ;
– une société d'agents $X_1^0$ créée à l'instant $t = 0$.

D'après ce le diagramme de temps, les agents clients $a_{1,1}$, $a_{2,1}$ et $a_{3,1}$ interagissent plusieurs fois avec la société $X_1^0$ entre les instants $t = 4s$, et $t = 15s$. Les agents $a_{1,1}$ et $a_{3,1}$ envoient deux premières demandes à $t = 4s$, à cet instant, la société s'active et commence à collecter les demandes pendant la période $Tacq = 3s$. Aux instants $t = 5s$ et $t = 6s$ les agents $a_{2,1}$ et $a_{3,1}$ envoient respectivement de nouvelles demandes. Ces deux dernières demandes sont traitées par la même société qui est encore disponible.

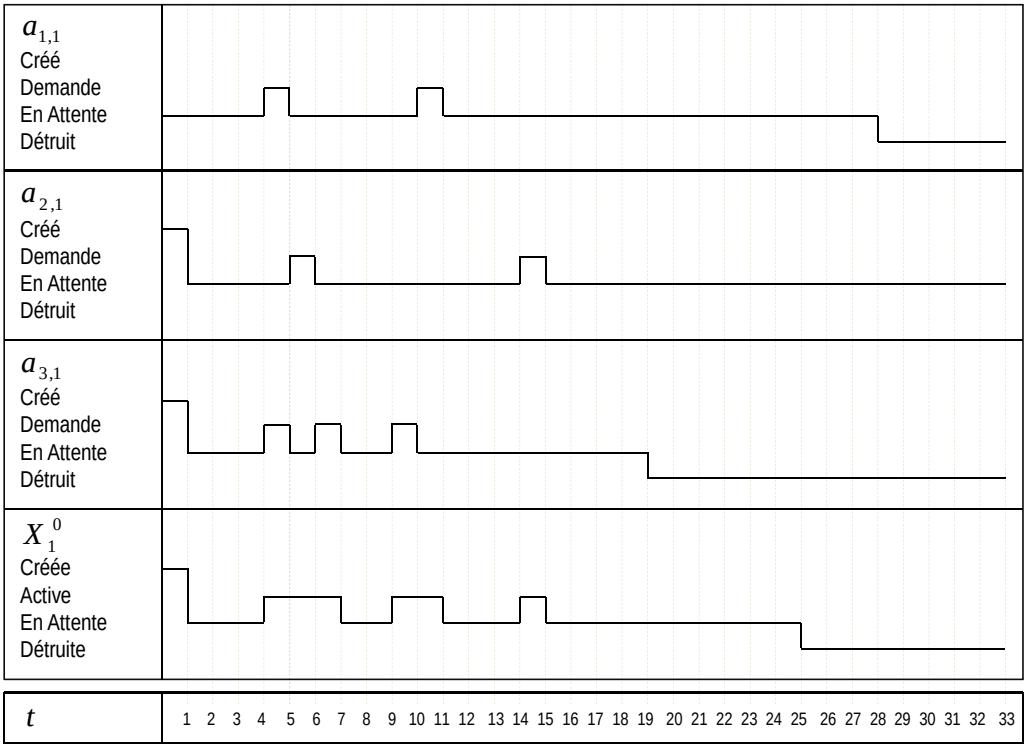



FIGURE III.9: Diagramme de temps de l'interaction d'une société d'agents avec 3 clients.

Le même scénario se déroule avec les deux demandes effectuées par les agents $a_{3,1}$ et $a_{1,1}$ aux instants respectifs $t = 9s$, et $t = 10s$. Puisque la durée $Tin = 10s$, la société $X_1^0$ est toujours disponible pour satisfaire la nouvelle demande effectuée par $a_{2,1}$ à l'instant $t = 14s$ car sa dernière activation est à $t = 11s$. A partir de $t = 15s$, la société $X_1^0$ ne reçoit plus de demandes jusqu'à l'instant $t = 25s$. Dans ce cas elle se termine puisque la durée $Tin = 10s$ est écoulée. Toute nouvelle demande de service formulée à partir de $t = 25s$ va déclencher la création d'une nouvelle société d'agents et le même cycle se répète.

### III.4.3 Discussion

La plateforme présentée, propose trois niveaux d'optimisation : 1) la recherche et la localisation des services, 2) le stockage des données et 3) la communication (i.e. les échanges à travers le réseau).

- $1^{er}$ *niveau. Recherche et localisation des services* : identifier les similarités dans les requêtes utilisateurs pour éviter les recherches redondantes des données ; optimiser les itinéraires des agents $Ar$ qui vont se déplacer dans le RDTC et collecter des données ; fournir des tables d'état aux agents $Au$ qui leurs permettront de faire des recherches locales et donc des recherches plus rapides ;
- $2^{ème}$ *niveau. Stockage des données* : hébergement de quelques services encore valides chez les membres d'un voisinage collaboratif ; stockage des données récurrentes (i.e. susceptibles d'être demandées plusieurs fois par les utilisateurs) dans l'entrepôt temporaire de données (ETD). (III.5.3) ;
- $3^{ème}$ *niveau. Communication* : le connecteur ne joue plus un rôle central dans la recherche et la distribution des IMA. Les utilisateurs sont capables d'échanger les données si possible d'une manière autonome sans l'intervention du connecteur. Cette communication est améliorée grâce au protocole de négociation multiobjectif, multilatéral (i.e. implique plusieurs agents) et à accords partiels que nous détaillons dans le chapitre IV.

Les agents $Ar$ sont responsable de la collecte des données à partir du RDTC afin de satisfaire les demandes utilisateurs. Le déplacement dans les nœuds du RDTC est guidé par les plans de route créés par l'agent $Ao$. Le paragraphe suivant présente l'approche d'optimisation intégrée dans un agent $Ao$ pour générer les itinéraires optimisés des agents $Ar$.

## III.5 Optimisation des Itinéraires des Agents Ramasseurs Mobiles

Afin de répondre aux requêtes utilisateurs, le connecteur dispose d'un ensemble d'agents mobiles (i.e. les agents ramasseurs) qui sont capables de se déplacer dans les nœuds du RDTC et collecter les données nécessaires pour générer les réponses. Chaque agent $Ar$ se déplace selon un itinéraire pré-calculé par l'agent $Ao$.

En effet, la génération des itinéraires des agents $Ar$ repose sur une méthode métaheuristique et plus précisément un algorithme génétique. Ce type d'algorithme que nous avons introduit dans le chapitre II, est convenable pour la résolution des problèmes NP-difficiles. Il est en particulier adapté à notre problématique et en adéquation avec l'architecture d'agents distribuée de notre système.

Dans cette partie, nous présentons un codage adapté à la problématique de génération d'un chromosome minimisant le coût total d'un ensemble de requêtes utilisateurs simultanées. Nous présentons aussi les opérateurs génétiques de croisement, de mutation et remplacement responsables de la génération des solutions possibles. Finalement, nous présentons les fonctions d'évaluation (appelées aussi fonction de fitness) responsables d'évaluer la pertinence des solutions générées selon les critères coût et délai de réponse.

### III.5.1 Formulation du Problème

Dans cette partie, nous évoquons la représentation du chromosome qui s'adapte bien avec le problème d'affectation des services aux fournisseurs potentiels. Le chromosome, noté par $H$, est représenté par une matrice à $M_j^t$ lignes et $V_j^t$ colonnes. Les lignes de la matrice représentent l'ensemble des services élémentaires demandés, tandis que les colonnes représentent les fournisseurs de services pouvant satisfaire ces demandes (faisant partie du RDTC).

La demande d'un service $s_m$ ($m \in [1, M_j^t]$), possédant un identifiant de service $ids_m \in \mathbb{S}_j^t$, à un instant $t$, peut générer trois cas de figure :
- le fournisseur de service $F_v \in \mathbb{F}_j^t$ ($v \in [1, V_j^t]$) est sélectionné pour satisfaire le service $s_m$ ;
- $F_v$ n'est pas sélectionné mais il peut quand même satisfaire le service $s_m$ ;
- $F_v$ ne peut pas satisfaire le service $s_m$.

**Hypothèse 1**. Nous supposons que $\forall ids_m \in \mathbb{S}_j^t$, $\exists F_v \in \mathbb{F}_j^t$ capable de satisfaire le service $s_m$ correspondant.

La génération des solutions revient dans ce cas à affecter chaque service $s_{i\in[1,M_j^t]}$ à un fournisseur de service $F_v$ tout en minimisant le coût et le délai de réponse de la solution globale. Par conséquent, nous adoptons le codage suivant :

$$H(m,v) = \begin{cases} 1 & \text{si } F_v \text{ est affecté à } s_m \\ * & \text{si } F_v \text{ peut être affecté à } s_m \\ 0 & \text{si } F_v \text{ ne peut pas être affecté à } s_m \end{cases} \quad \text{(III.4)}$$

Si un serveur $F_{v\in[1,V_j^t]}$ ne propose pas un service $ids_{m\in[1,M_j^t]}$ alors il ne peut pas affecté à ce dernier dans la matrice du chromosome $H$. Dans le tableau III.5 nous présentons un exemple d'un chromosome à 10 lignes et 15 colonnes. Selon cet exemple, les services $ids_{10}$, $ids_{32}$, $ids_4$, $ids_5$, $ids_{16}$, $ids_1$, $ids_{25}$,

$ids_{12}$, $ids_3$ et $ids_{17}$ sont affectés respectivement aux serveurs $F_2$, $F_{22}$, $F_{10}$, $F_{12}$, $F_{12}$, $F_7$, $F_9$, $F_8$, $F_{32}$ et $F_3$.

Nous remarquons aussi que les serveurs $F_{42}$, $F_{10}$, $F_{11}$, $F_2$, $F_{18}$, $F_{71}$ et $F_{32}$ sont capables de fournir le service $ids_{10}$ mais seulement le serveur $F_2$ a été choisi ; les autres serveurs ne proposent pas ce service.

Les algorithmes des opérateurs génétiques manipulant ce chromosome ainsi que les algorithmes d'évaluation des solutions sont proposés dans le paragraphe suivant.

TABLE III.5: Exemple d'un chromosome de 10 lignes et 15 colonnes.

| | $F_3$ | $F_7$ | $F_{42}$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_2$ | $F_{63}$ | $F_8$ | $F_9$ | $F_{18}$ | $F_{71}$ | $F_{22}$ | $F_{32}$ | $F_{40}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $ids_{10}$ | 0 | 0 | * | * | * | 0 | 1 | 0 | 0 | 0 | * | * | 0 | * | 0 |
| $ids_{32}$ | * | * | 0 | 0 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | * | * |
| $ids_4$ | * | * | 0 | 1 | 0 | * | * | 0 | * | 0 | 0 | * | 0 | * | * |
| $ids_5$ | 0 | * | 0 | * | * | 1 | 0 | * | * | 0 | * | * | 0 | 0 | * |
| $ids_{16}$ | * | 0 | * | 0 | 0 | 1 | * | * | 0 | * | 0 | 0 | * | * | * |
| $ids_1$ | 0 | 1 | 0 | * | 0 | 0 | 0 | * | * | 0 | * | 0 | * | * | 0 |
| $ids_{25}$ | * | * | * | 0 | * | 0 | * | 0 | 0 | 1 | 0 | * | 0 | 0 | * |
| $ids_{12}$ | 0 | * | 0 | * | 0 | 0 | 0 | * | 1 | 0 | * | 0 | * | * | 0 |
| $ids_3$ | 0 | 0 | * | 0 | * | * | * | 0 | * | 0 | * | * | * | 1 | 0 |
| $ids_{17}$ | 1 | 0 | * | * | 0 | 0 | * | * | 0 | * | * | 0 | * | 0 | * |

### III.5.2 Génération des Itinéraires des Agents Ramasseurs Mobiles

Afin de satisfaire les demandes des utilisateurs, le connecteur dispose d'un ensemble d'agents patrouilleurs (i.e. les agents $Ar$) prêts à se déplacer dans les nœuds du RDTC et collecter les données demandées. Chaque agent $Ar$ est responsable de la visite d'un ensemble de nœuds dans le RDTC à partir d'un nœud de départ noté par $D_{l\in[1,L]}$ (Fig. III.10). C'est à dire que chaque connecteur possède son propre départ d'agents mobiles dans le RDTC.

Pour cela, l'agent ramasseur ($Ar$) a besoin d'un ensemble d'informations de routage que nous appelons itinéraire ou plan de route. Grâce à un algorithme d'optimisation génétique que nous présentons dans cette partie, le connecteur génère à chaque période $Tacq$ des itinéraires optimisés aux agents $Ar$.

Le paragraphe suivant présente les différents algorithmes d'optimisation génétique développés pour la génération des chromosomes optimisés utilisés pour élaborer les itinéraires des agents ramasseurs à travers le RDTC. A la fin de ce paragraphe, nous présentons un tableau récapitulatif des différentes variables utilisées.

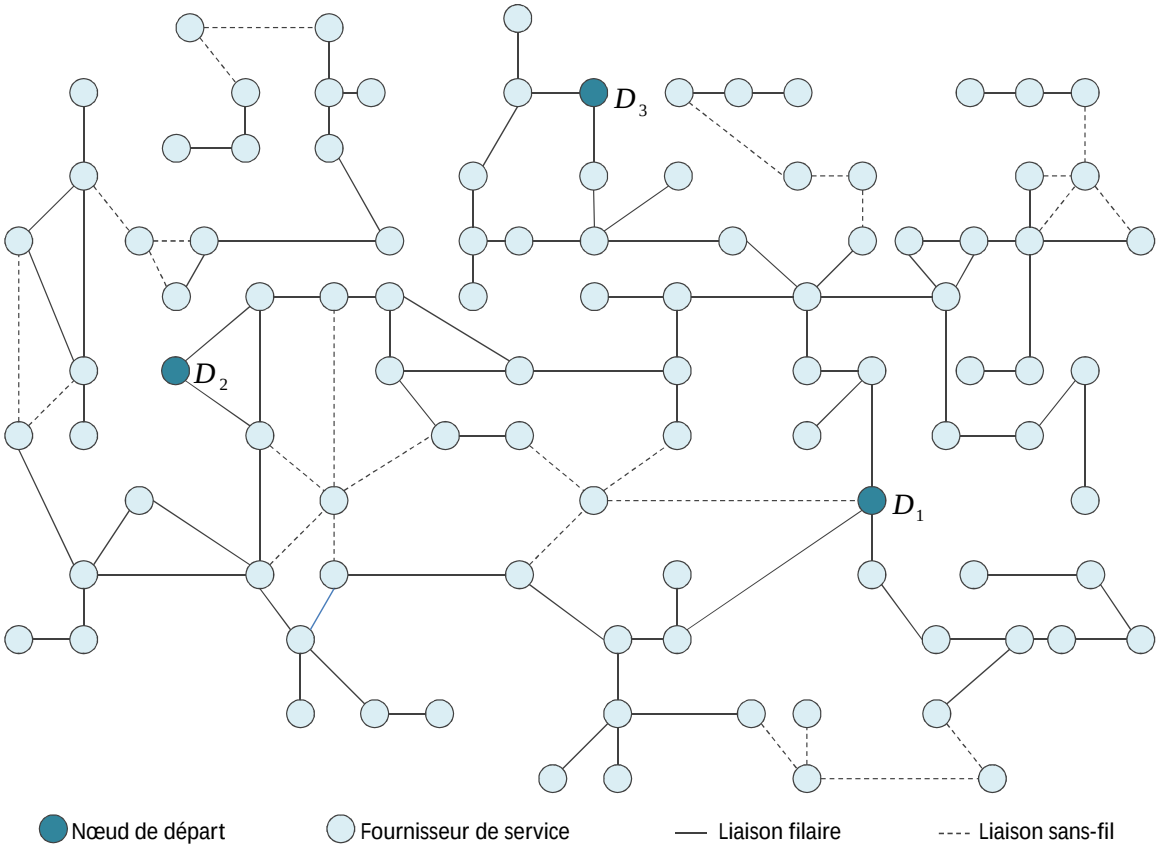



FIGURE III.10: Exemple d'un réseau distribué de fournisseurs de services de transport comodal avec trois départs $D_1$, $D_2$ et $D_3$.

#### III.5.2.1 Algorithmes d'Optimisation Développés

Dans ce qui suit, nous présentons les algorithmes génétiques conçus. L'approche d'optimisation proposée se compose de deux opérateurs génétiques :

– un opérateur de croisement/correction ;
– et un opérateur de mutation.

Dans ce travail, nous proposons l'optimisation d'un système d'information d'aide à la mobilité en associant les algorithmes génétiques à un système d'agents. L'objectif de cette thèse n'est pas donc le test des méthodes d'optimisation. Cependant, nous nous intéressons à concevoir et optimiser ce système pour qu'il soit en mesure de gérer un nombre élevé de demandes simultanées tout en optimisant les coûts, les temps de réponses et les quantités de données transférées des services demandés.

**a) Création de la population initiale** (Algorithme 2). Nous nous proposons de créer la population initiale $\mathbb{P}_0$ par la construction aléatoire d'un ensemble de $N$ solutions. L'algorithme 2 illustre la création de l'ensemble $\mathbb{P}_0$.

**Algorithme 2** Création de la population initiale

**Entrée:** $\mathbb{F}_j^t$, $\mathbb{S}_j^t$, nombre d'individus $N$, Table de services $Tab$
**Sortie:** $\mathbb{P}_0$

$M_j^t \leftarrow Cardinal(\mathbb{S}_j^t)$, $V_j^t \leftarrow Cardinal(\mathbb{F}_j^t)$
Initialiser une matrice $H(M_j^t \times V_j^t)$
Initialiser une liste $list \leftarrow null$
Initialiser un compteur $k \leftarrow 1$
**pour** $i = 1$ jusqu'à $N$ **faire**
  **pour** $m = 1$ jusqu'à $M_j^t$ **faire**
    **pour** $v = 1$ jusqu'à $V_j^t$ **faire**
      **si** $Tab[m, v] \neq -$ **alors**
        $H[m, v] \leftarrow *$
        $list[k] \leftarrow v$
        $k \leftarrow k + 1$
      **sinon**
        $H[m, v] \leftarrow 0$
      **fin si**
    **fin pour**
    Sélectionner aléatoirement un élément $c$ de $list$ $//c \in [1, k-1]$
    $H[m, c] \leftarrow 1$
  **fin pour**
**fin pour**
**retourner** $\mathbb{P}_0$

**b) Étape de croisement** (Algorithme 3). L'étape de croisement/correction contrôlée est illustrée par l'algorithme 3. Au cours de cette étape, l'algorithme procède tout d'abord par une sélection par tournoi avec un taux de croisement $\lambda_c \in ]0, 1[$. La taille adoptée du tournoi est égale à $n_c$, avec $n_c \geq 2$. Ensuite, il sélectionne avec $\lambda_c$ aussi un fournisseur de service : $F_v \in \mathbb{F}_j^t$.

Après, l'algorithme crée un ensemble de $n_c$ nouveaux chromosomes enfants $\{E_1, E_2, .., E_{n_c}\}$, en faisant en sorte que la colonne correspondant au serveur $F_v$ dans chacun des chromosomes parents $\{P_1, P_2, .., P_{n_c}\}$ correspondants soit affectée respectivement aux éléments de l'ensemble des chromosomes enfants.

Par la suite, il effectue une rotation d'un pas à droite des chromosomes parents et affecte les colonnes restantes aux enfants correspondants, c'est à dire, les colonnes de $\{P_{n_c}, P_1, .., P_{n_c-1}\}$ sont affectées respectivement à $\{E_1, E_2, .., E_{n_c}\}$.

Une solution n'est considérée viable que quand un service demandé n'est affecté qu'à un seul fournisseur de service dans le chromosome. Toute autre situation est considérée comme non viable et le chromosome en question doit passer par une étape de correction.

Les tables III.6 et III.7 illustrent un exemple pour une taille de tournoi $n_c = 3$ et un fournisseur de service $F_3$. Il est clair que les enfants $E_1$ et $E_3$ ne sont pas des solutions viables et seront corrigé dans

une prochaine étape. En effet, dans $E_1$, aucun fournisseur de service n'est affecté au service $ids_1$. Dans $E_3$, les fournisseurs $F_2$ et $F_3$ sont affectés au même service $ids_1$.

TABLE III.6: Exemple de trois chromosomes parents

| $P_1$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | 1 | $*$ | $*$ |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | $*$ | 1 | 0 |

| $P_2$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | $*$ | 1 | $*$ |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | 1 | $*$ | 0 |

| $P_3$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | $*$ | $*$ | 1 |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | 1 | $*$ | 0 |

TABLE III.7: Exemple de trois chromosomes enfants

| $E_1$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | $*$ | $*$ | $*$ |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | 1 | $*$ | 0 |

| $E_2$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | 1 | $*$ | $*$ |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | $*$ | 1 | 0 |

| $E_3$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | $*$ | 1 | 1 |
| $ids_2$ | 1 | 0 | 0 |
| $ids_3$ | 1 | $*$ | 0 |

L'étape de correction consiste à corriger des éventuelles erreurs survenues au cours de l'étape précédente. En effet, deux cas de figures où la solution est considérée comme non viable peuvent se produire :

– un service n'est affecté à aucun fournisseur de service
– un service est affecté à plus qu'un fournisseur de service

Au départ, l'algorithme récupère les $n_c$ individus générés dans l'étape précédente. Pour chaque individu, il parcourt ligne par ligne le chromosome et fait en sorte que la solution soit viable. Il faut remarquer que même lorsqu'une solution est viable à l'entrée, elle peut ne pas être la même à la sortie. Cependant, toutes les sorties sont toujours des solutions viables.

**c) Étape de mutation** (Algorithme 4). Dans cette étape, l'algorithme 4 procède à une mutation par échange avec un taux de mutation $\lambda_m \in ]0, 1[$. L'objectif de cet opérateur est de varier la population courante afin d'améliorer la diversité des individus et éviter au maximum toute convergence prématurée.

**d) Étape d'Évaluation** (Algorithme 5). Dans cette étape, l'objectif est de choisir le meilleur chromosome de la population. Nous proposons deux critères d'optimisation : le coût et la quantité de données normalisés de la solution. Rappelons que le coût normalisé correspondant à $c_{m,v}$ est notée par $\bar{c}_{m,v}$ et que la quantité de données moyenne normalisée correspondante à $q_{m,v}$ est notée par $\bar{q}_{m,v}$.

Nous supposons que les coûts et les quantités de données normalisés sont regroupés dans une table notée par $\overline{CQ}$. La table III.8 illustre un exemple d'une table de coûts/quantités de données normalisés $\overline{CQ}$ avec $\mathbb{F}_j^t = \{F_1, F_2, F_3\}$ et $\mathbb{S}_j^t = \{ids_1, ids_2, ids_3\}$. $\overline{CQ}$ est composée de pairs

**Algorithme 3** Algorithme de l'opérateur de croisement/correction

**Entrée:** Ancienne génération $\mathbb{P}$, $n_c$, $\mathbb{F}_j^t$, $\mathbb{S}_j^t$, taux de croisement $\lambda_c$
**Sortie:** Nouvelle génération $\mathbb{E}$

//Étape 1 : Croisement
Sélectionner aléatoirement avec $\lambda_c$ $n_c$ individus de $\mathbb{P}$ : $P_1, P_2, .., P_{n_c}$
Initialiser un vecteur $v_p \leftarrow [P_1, P_2, .., P_{n_c}]$
Sélectionner aléatoirement avec $\lambda_c$ un fournisseur de service de $\mathbb{F}_j^t$ : $F_v$
Initialiser $n_c$ nouveaux individus dans un vecteur : $v_e \leftarrow [E_1, E_2, .., E_{n_c}]$
$v_e \leftarrow v_p$
Faire une rotation à droite de $v_p$
**pour** $i = 1$ jusqu'à $n_c$ **faire**
  **pour** $j = 1$ jusqu'à $Cardinal(\mathbb{F}_j^t)$ **faire**
    **si** $j^{ème}$ *colonne de* $v_e[i]$ ne correspond pas à $F_v$ **alors**
      $j^{ème}$ *colonne de* $v_e[i] \leftarrow j^{ème}$ *colonne de* $v_p[i]$
    **fin si**
  **fin pour**
**fin pour**
//Étape 2 : Correction
Initialiser une liste $list$
Initialiser un compteur $u \leftarrow 1$
**pour** $i = 1$ jusqu'à $n_c$ **faire**
  $e \leftarrow v_e[i]$
  **pour** $k = 1$ jusqu'à $Cardinal(\mathbb{S}_j^t)$ **faire**
    **pour** $l = 1$ jusqu'à $Cardinal(\mathbb{F}_j^t)$ **faire**
      **si** $e[k, l] = 1$ ou $e[k, l] = *$ **alors**
        $list[u] \leftarrow l$
        $u \leftarrow u + 1$
      **fin si**
    **fin pour**
    **si** $Longueur(list) = 1$ **alors**
      $e[k, list[1]] \leftarrow 1$
    **sinon**
      **pour** $z = 1$ jusqu'à $Longueur(list)$ **faire**
        $e[k, z] \leftarrow *$
      **fin pour**
      Sélectionner aléatoirement un élément $c$ de $list$ $//c \in [1, k-1]$
      $e[k, c] \leftarrow 1$
    **fin si**
  **fin pour**
  $v_e[i] \leftarrow e$
**fin pour**
Substituer $v_p$ par $v_e$ dans $\mathbb{P}$ et affecter le résultat à $\mathbb{E}$
**retourner** $\mathbb{E}$

**Algorithme 4** Algorithme de l'opérateur de mutation adopté

**Entrée:** Ancienne génération $\mathbb{P}$, $\mathbb{F}_j^t$, $\mathbb{S}_j^t$, taux de mutation $\lambda_m$
**Sortie:** Nouvelle génération $\mathbb{E}$

```
Sélectionner aléatoirement avec λ_m un individu de ℙ : P_a
Sélectionner aléatoirement avec λ_m un service de 𝕊_j^t : ids_m
Initialiser une liste list de longueur Cardinal(𝔽_j^t)
Initialiser un variable b ← indice de ids_m dans P_a
Initialiser une variable c
Initialiser un compteur k ← 1
pour i = 1 jusqu'à Cardinal(𝔽_j^t) faire
    si P_a[b, i] égal à ∗ alors
        list[k] ← i
        k ← k + 1
    sinon si P_a[b, i] égal à 1 alors
        c ← i
    fin si
    Sélectionner aléatoirement un élément d de list
    Permuter P_a[b, c] et P_a[b, d]
fin pour
retourner 𝔼
```

$\{\overline{CQ}.cout, \overline{CQ}.quantite\}$ où $\overline{CQ}.cout$ et $\overline{CQ}.quantite$ correspondent respectivement aux coût quantité de données normalisés d'un service $ids_m$ proposé par un serveur $F_v$.

TABLE III.8: Exemple d'une table de coûts/quantités de données normalisés.

| | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| $ids_1$ | $\{0,48; 0,84\}$ | $\{0,31; 0,42\}$ | $\{0,55; 0,64\}$ |
| $ids_2$ | $\{0,13; 0,57\}$ | $\{0,09; 0,19\}$ | $\{0,70; 0,33\}$ |
| $ids_3$ | $\{0,62; 0,90\}$ | $\{0,41; 0,43\}$ | $\{0,63; 0,85\}$ |

L'algorithme d'évaluation effectue une agrégation du coût et de la quantité de données normalisés des services de chaque chromosome. Pour cela, deux poids $w_c$ et $w_q$ sont associés respectivement aux coût et quantité de données, avec $w_c + w_q = 1$. L'algorithme illustre l'étape d'évaluation des solutions.

TABLE III.9: Récapitulatif des variables du système

| **Notation** | **Description** |
|---|---|
| $\lambda_c$ | probabilité de croisement |
| $\lambda_m$ | probabilité de mutation |
| $\overline{CQ}$ | table des coûts et des quantités de données normalisés |
| $\mathbb{E}$ | génération des enfants (nouvelle génération) |
| $E*$ | un chromosome optimal |
| $N$ | nombre total d'individus dans une population |
| $n_c$ | taille d'un tournoi ou nombre des individus à croiser |
| $\mathbb{P}$ | génération des parents (ancienne génération) |
| $\mathbb{P}_0$ | population initiale |

**Algorithme 5** Algorithme d'évaluation des chromosomes

**Entrée:** Nouvelle génération $\mathbb{E}$, $\mathbb{F}_j^t$, $\mathbb{S}_j^t$, $w_c$, $w_q$ et $\overline{CQ}$
**Sortie:** Solution $E*$

$M_j^t \leftarrow Cardinal(\mathbb{S}_j^t)$, $V_j^t \leftarrow Cardinal(\mathbb{F}_j^t)$
Initialiser une matrice $H(M_j^t \times V_j^t)$
Initialiser une table $tabEval$ de longueur $Cardinal(\mathbb{E})$
**pour** $i = 1$ jusqu'à $Cardinal(\mathbb{E})$ **faire**
  Initialiser une variable $cout \leftarrow 0$
  Initialiser une variable $quantite \leftarrow 0$
  **pour** $k = 1$ jusqu'à $M_j^t$ **faire**
    Initialiser une variable booléenne $drapeau \leftarrow vrai$
    Initialiser un compteur $l \leftarrow 1$
    **tant que** $drapeau$ **faire**
      **si** $H[k, l] = 1$ **alors**
        $drapeau \leftarrow faux$
        $cout \leftarrow cout + \overline{CQ}.cout[k, l]$
        $quantite \leftarrow quantite + \overline{CQ}.quantite[k, l]$
      **fin si**
      $l \leftarrow l + 1$
    **fin tant que**
  **fin pour**
  $tabEval[i] \leftarrow w_c\ cout + w_q\ quantite$
**fin pour**
$E* \leftarrow$ chromosome de $\mathbb{E}$ qui correspond à $\min(tabEval)$
**retourner** $E*$

#### III.5.2.2 Répartition du Réseau des Fournisseurs de Services

Dans cette partie, nous évoquons les développement de l'approche d'évaluation des solutions générées par les algorithmes présentés dans la partie précédente. Nous présentons ainsi l'élaboration finale des itinéraires des agents $Ar$.

**a) Hypothèses et Définitions**

Comme le présente la Fig. III.10, les connecteurs répartis dans plusieurs espaces ubiquitaires possèdent chacun un départ différent pour les agents $Ar$. L'ensemble des agents $Ar$ qu'un connecteur $C_j$ peut générer est noté par $AR_j = \{Ar_{1,j}, Ar_{2,j}, .., Ar_{k,j}\}$. En supposant que le RDTC est un réseau distribué, la collecte des données doit être optimisée afin de satisfaire les contraintes temporelles sur les délais de réponse des requêtes utilisateurs. Pour cela, le connecteur doit prendre en considération ces contraintes lors de la génération des plans de route des agents mobiles. Dans ce qui suit, nous présentons une modélisation du RDTC et des flux de transfert des données dans le réseau. Nous rappelons que $V_j^t$ est le cardinal de l'ensemble des fournisseurs de services sélectionnés pour répondre aux requêtes utilisateurs reçues par un connecteur $C_j$ à un instant $t$. Commençons par définir les paramètres suivants :

**Hypothèse 2**. Nous supposons que tous les agents $Ar$ possèdent une quantité de données initiale noté par $q_0$ qui correspond à leurs propres codes et données internes.

**Poids** ($p_{k,j,v}$) : un Poids $p_{k,j,v}$ d'un agent $Ar_{k,j}$ se déplaçant vers un nœud $F_v$ représente la somme des quantités des données collectées au cours de son chemin jusqu'à ce nœud. Rappelons que tout résultat de l'invocation d'un service $ids_{m\in[1,M_j^t]}$ proposé par un serveur $F_v$ possède une quantité de données noté par $q_{m,v}$. Le poids est calculé comme suit :

$$p_{k,j,v} = q_0 + \sum_{x=1}^{v} q_{m,x} \tag{III.5}$$

Si plusieurs services sont collectés depuis le même serveur, il suffit de remplacer $q_{m,x}$ (resp. $te_{m,x}$) par $\sum_m q_{m,x}$ (resp. $\sum_m te_{m,x}$). Ceci ne posera pas de problème aux équations ci-dessous puisqu'elles ne sont pas en fonction de $m$.

**Hypothèse 3**. Nous supposons que le débit de transmission des données d'un nœud $F_v$ vers un nœud $F_{v'}$, noté par $d(F_v, F_{v'})$, est accessible (ex. RSSI [4]).

**Hypothèse 4**. Nous supposons que le débit de transmission des données entre deux nœuds $F_v$ et $F_{v'}$ ne dépend pas du sens (i.e. ascendant ou descendant). Autrement dit :

$$d(F_v, F_{v'}) = d(F_{v'}, F_v) \tag{III.6}$$

**Temps de transmission** ($tr_{k,j,v,v'}$) : le temps de transmission est la durée nécessaire pour qu'un agent $Ar_{k,j}$ de poids $p_{k,j,v}$ se déplace d'un nœud $F_v$ vers un nœud $F_{v'}$. Le temps de transmission est calculé comme suit :

$$tr_{k,j,v,v'} = \frac{p_{k,j,v}}{d(F_v, F_{v'})} = \frac{q_0 + \sum_{x=1}^{v} q_{m,x}}{d(F_v, F_{v'})} \tag{III.7}$$

**Temps de parcours** ($tp_{k,j}$) : le temps de parcours est la durée totale d'un plan de route $\{D_j, F_1, .., F_{V_j^t}\}$ d'un agent $Ar_{k,j}$ (rappelons que $D_j$ est le nœud de départ des agents ramasseurs appartenant à un connecteur $C_j$). Le temps de parcours est calculé comme suit :

$$\begin{aligned} tp_{k,j} = & \ \frac{q_0}{d(D_j,F_1)} + \sum_{x=1}^{V_j^t-1} tr_{k,j,x,x+1} + \sum_{x=1}^{V_j^t} te_{m,x} + \frac{q_0+\sum_{x=1}^{V_j^t} q_{m,x}}{d(F_{V_j^t},D_j)} \\ = & \ \frac{q_0}{d(D_j,F_1)} + \sum_{x=1}^{V_j^t-1} [\frac{q_0+\sum_{y=1}^{x} q_{m,y}}{d(F_x,F_{x+1})}] + \sum_{x=1}^{V_j^t} te_{m,x} + \frac{q_0+\sum_{x=1}^{V_j^t} q_{m,x}}{d(F_{V_j^t},D_j)} \end{aligned} \tag{III.8}$$

4. Received Signal Strength Indicator (indicateur de force du signal reçu)

Afin de simplifier cette équation nous supposons que la quantité $q_0 = 0$ et que le taux de transfert entre les différents pairs de serveurs est une constante égale à $d$. Dans ce cas, l'équation III.8 devient :

$$\begin{aligned} tp_{k,j} = & \ \frac{1}{d}\sum_{x=1}^{V_j^t-1}\sum_{y=1}^{x} q_{m,y} + \sum_{x=1}^{V_j^t} te_{m,x} + \frac{1}{d}\sum_{x=1}^{V_j^t} q_{m,x} \\ = & \ \frac{1}{d}\sum_{x=1}^{V_j^t}\sum_{y=1}^{x} q_{m,y} + \sum_{x=1}^{V_j^t} te_{m,x} \end{aligned} \tag{III.9}$$

d'où

$$tp_{k,j} = \sum_{x=1}^{V_j^t}[\frac{1}{d}\sum_{y=1}^{x} q_{m,y} + te_{m,x}] \tag{III.10}$$

**Date de réponse au plus tard** ($Dmax_j^t$) : la date de réponse au plus tard d'une solution $H$ à l'instant $t$ est égale à la date la plus faible des dates de réponses au plus tard de l'ensemble des requêtes simultanées $R_j^t$ : $Dmax_j^t = min(dmax_{h,j})_{h\in[1,Cardinal(R_j^t)]}$.

Le temps de parcours que nous venons de calculer dans l'équation III.9 peut dépasser cas le délai de réponse au plus tard fixé par le connecteur pour cet ensemble de services. Pour résoudre ce problème, il faut construire les plans de routes (i.e. les itinéraires) de façon à ce que l'agent $Ar$ puisse se déplacer, collecter les données et retourner au nœud de départ sans dépasser la date de réponse au plus tard $Dmax_j^t$. Dans la partie suivante, nous présentons l'algorithme de répartition du RDTC qui permet d'assurer cette contrainte temporelle pour chaque agent $Ar$.

**b) Algorithme de Répartition**

La répartition du RDTC consiste à attribuer les nœuds à visiter à un ensemble d'agents ramasseurs mobiles. Ces nœuds représentent la solution issue du processus d'optimisation (i.e. l'ensemble des nœuds à visiter et les services à collecter sur ces nœuds) de manière à ce que les dates de réponse au plus tard soient toujours respectées. Si nous reprenons l'exemple de la table III.5, une subdivision possible du chromosome s'effectue comme le montre la table III.10, où chaque section représentée par une couleur différente est associée à un agent $Ar$ différent. Dans cet exemple, trois agents ramasseurs $Ar_{1,1}$, $Ar_{2,1}$ et $Ar_{3,1}$ appartenant à un connecteur $C_1$, doivent collecter respectivement les ensembles de services suivants : $\{ids_{10}, ids_{32}, ids_4, ids_5, ids_{16}\}$, $\{ids_1, ids_{25}, ids_{12}\}$ et $\{ids_3, ids_{17}\}$.

La répartition du RDTC selon $\mathbb{S}_j^t$ et $\mathbb{F}_j^t$, illustrée par l'algorithme 6, s'effectue par le calcul du temps de parcours $tp_{k,j}$ d'un agent $Ar_{k,j}$ pour différents itinéraires possibles et choisir l'itinéraire qui permet de récupérer le maximum de données possible sans dépasser la date de réponse au plus tard définie comme suit :

Le résultat de l'algorithme est une table appelée *table d'itinéraires* qui définit le nombre d'agents ramasseurs $Ar$ à créer ainsi que la liste des nœuds à visiter par chaque agent $Ar$. La table III.11

TABLE III.10: Exemple de répartition d'un chromosome de 10 lignes et 15 colonnes.

| | $F_3$ | $F_7$ | $F_{42}$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_2$ | $F_{63}$ | $F_8$ | $F_9$ | $F_{18}$ | $F_{71}$ | $F_{22}$ | $F_{32}$ | $F_{40}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $ids_{10}$ | 0 | 0 | $*$ | $*$ | $*$ | 0 | 1 | 0 | 0 | 0 | $*$ | $*$ | 0 | $*$ | 0 |
| $ids_{32}$ | $*$ | $*$ | 0 | 0 | $*$ | $*$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | $*$ | $*$ |
| $ids_4$ | $*$ | $*$ | 0 | 1 | 0 | $*$ | $*$ | 0 | $*$ | 0 | 0 | $*$ | 0 | $*$ | $*$ |
| $ids_5$ | 0 | $*$ | 0 | $*$ | $*$ | 1 | 0 | $*$ | $*$ | 0 | $*$ | $*$ | 0 | 0 | $*$ |
| $ids_{16}$ | $*$ | 0 | $*$ | 0 | 0 | 1 | $*$ | $*$ | 0 | $*$ | 0 | 0 | $*$ | $*$ | $*$ |
| $ids_1$ | 0 | 1 | 0 | $*$ | 0 | 0 | 0 | $*$ | $*$ | 0 | $*$ | 0 | $*$ | $*$ | 0 |
| $ids_{25}$ | $*$ | $*$ | $*$ | 0 | $*$ | 0 | $*$ | 0 | 0 | 1 | 0 | $*$ | 0 | 0 | $*$ |
| $ids_{12}$ | 0 | $*$ | 0 | $*$ | 0 | 0 | 0 | $*$ | 1 | 0 | $*$ | 0 | $*$ | $*$ | 0 |
| $ids_3$ | 0 | 0 | $*$ | 0 | $*$ | $*$ | $*$ | 0 | $*$ | 0 | $*$ | $*$ | $*$ | 1 | 0 |
| $ids_{17}$ | 1 | 0 | $*$ | $*$ | 0 | 0 | $*$ | $*$ | 0 | $*$ | $*$ | 0 | $*$ | 0 | $*$ |

illustre un exemple d'une table d'itinéraires.

TABLE III.11: Exemple d'une table d'itinéraire avec 3 agents $Ar$

| | $F_3$ | $F_7$ | $F_{42}$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_2$ | $F_{63}$ | $F_8$ | $F_9$ | $F_{18}$ | $F_{71}$ | $F_{22}$ | $F_{32}$ | $F_{40}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Ar_{1,1}$ | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $Ar_{2,1}$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $Ar_{3,1}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Les informations récupérées par les agents $Ar$ depuis le RDTC peuvent être classées et stockées dans une base de données appelée Entrepôt Temporaire de Données (ETD) dans le but d'éviter la recherche redondante des informations pendant des différentes périodes d'acquisition. Dans ce qui suit, nous présentons l'ETD.

### III.5.3 Classification et Stockage des Données

Nous avons développé une méthode à base d'indices pour classer les informations (i.e. les IMA). En effet, les IMA proposées par les fournisseurs de services du RDTC peuvent être classées selon leur aspect temporel. En effet, certaines informations sont dynamiques et nécessites des mises à jours récurrentes. D'autres informations sont plutôt statiques et peuvent être stockées dans l'ETD afin de les réutiliser pour répondre aux nouvelles requêtes utilisateurs.

Pour cela, nous définissons trois classes d'informations et nous associons un indice différent à chaque classe afin de déterminer la décision des agents $Ar$ lors de la récupération d'un ensemble de données. La table III.12 présente les 3 classes ainsi que les indices et les décisions des agents $Ar$ associés à chaque classe.

**Algorithme 6** Algorithme de répartition du RDTC selon $\mathbb{S}'$ et $\mathbb{F}'$

**Entrée:** Solution $H$, $\mathbb{F}_j^t$, $\mathbb{S}_j^t$, $Dmax_j^t$
**Sortie:** Table d'itinéraires $tabItineraire$

```
1: Initialiser une matrice tabItineraire à zéro
2: Initialiser un compteur l ← 1
3: pour i = 1 jusqu'à Cardinal(S_j^t) faire
4:    Initialiser un compteur k ← 1
5:    Initialiser une variable booléenne drapeau ← vrai
6:    tant que drapeau faire
7:       si H[i,k] = 1 alors
8:          drapeau ← faux
9:          Calculer le temps de parcours tp jusqu'au serveur F_j^t[k]
10:         si tp < Dmax_j^t alors
11:            tabItineraire[l,k] ← 1
12:         sinon
13:            l ← l + 1
14:         fin si
15:      fin si
16:      k ← k + 1
17:   fin tant que
18: fin pour
19: retourner tabItineraire
```

Table III.12: Classes de données et décision d'un agent ramasseur

| **Classe de l'information** | **Indice** | **Décision de l'agent *Ar*** |
|---|---|---|
| Statique | S | Stocker l'information dans l'ETD |
| Très dynamique | T | Ne rien faire |
| Dynamique | D | Stocker l'information dans l'ETD |

L'ETD joue le rôle d'une mémoire tampon. En effet, quand un ensemble de requêtes simultanées est acquis par le connecteur, l'agent $Ad$ vérifie tout d'abord s'il existe des services demandés qui sont stockés dans l'ETD et encore valides. Dans le cas où il existe des requêtes (i.e. ensemble de services demandés) qui peuvent être satisfaites directement à partir de l'ETD, l'agent $Ad$ demande à l'agent $Ac$ de générer les réponses correspondantes. Dans le cas contraire, le connecteur lance la recherche comme expliqué précédemment.

A la fin de cette recherche, les agents $Ar$ retournent au connecteur et mettent à jour l'ETD selon l'indice de chaque service. Les services qui possèdent un indice T, c'est-à-dire il sont très dynamiques, ne peuvent pas être stockés dans cette base de données.

Cette même technique d'indices est utilisée par le connecteur pour déterminer si un client peut être autorisé à fournir un service donné ou non. En effet, seulement dans le cas où le service possède un indice D ou S, les agents $Au$ sont capable d'obtenir une permission de changement de rôle. Cette partie sera détaillée dans le prochain chapitre.

## III.6 Conclusion

Dans ce chapitre, nous avons présenté une architecture multi-agents d'un système d'information responsable de la recherche et la distribution des IMA d'une manière flexible. L'architecture est composée de deux sous-systèmes : les connecteurs et les voisinages collaboratifs. Le connecteur représente une interface entre les utilisateurs et le RDTC qui permet d'optimiser la recherche des services par le moyen d'un processus de décomposition-optimisation-collecte-composition. Les données sont collectées grâce à l'emploi des agents mobiles qui se déplacent selon des plans de route optimisés entre les nœuds du RDTC. La génération des itinéraires de ces agents est basée sur une approche d'optimisation génétique implantée dans le comportement du connecteur.

Grâce à une approche orientée rôle, les membres d'un voisinage collaboratif sont capables d'acquérir des rôles actifs et de fournir des informations pendant des durées de temps bien déterminées afin de les échanger les uns avec les autres. Cette approche permet de réduire la charge du connecteur et par conséquent améliorer la robustesse du système et sa capacité à supporter la montée en charge. Dans le chapitre suivant, nous présentons le paradigme de changement dynamique du rôle des agents utilisateurs ainsi que le protocole de négociation multilatéral, multicritère et à accord partiel.

# Chapitre IV

# Protocole de Négociation pour Optimiser les Communications Agents du Système Proposé

**Sommaire**

## IV.1 Introduction

Dans le chapitre précédent, nous avons présenté l'architecture du système proposé. Cette dernière est composée de deux sous-systèmes (connecteur et voisinage collaboratif) qui coopèrent afin d'optimiser la recherche, la collecte et la distribution des services. Nous avons détaillé les comportements des agents dans le connecteur et nous avons présenté l'approche d'optimisation adoptée. Dans ce chapitre, nous présentons les comportements des agents utilisateurs qui composent le voisinage collaboratif. Ces derniers sont capables de changer de rôle dynamiquement et devenir à leur tour des fournisseurs de services pour optimiser le temps de réponse et assister le connecteur à faire face à la montée en charge. Dans ce cas, les informations hébergées par des clients actifs peuvent être échangées ("vendues" et "achetées") comme dans une sorte de cybermarché.

Ce chapitre répond principalement à la question : *qui* change de rôle et pour fournir *quoi*, *comment* et pour combien de *temps* ? En effet, le connecteur est responsable d'attribuer des permissions aux agents utilisateurs pour changer leurs rôles et fournir certains services. La stratégie de changement dynamique du rôle implémentée par le connecteur, met en œuvre des indices qui permettent de maintenir un niveau suffisant de l'offre dans le voisinage collaboratif. Les agents utilisateurs sont donc capables de localiser les services dans le voisinage et les récupérer par le moyen d'un protocole de négociation innovant.

Ce chapitre est organisé en quatre parties : la première partie présente le comportement d'un agent utilisateur au sein d'un voisinage collaboratif. La deuxième partie présente l'algorithme d'administration du changement dynamique des rôles des agents du voisinage collaboratif et l'explique avec un exemple. La troisième partie détaille l'algorithme d'identification des fournisseurs de services dans un voisinage collaboratif et l'explique avec un exemple. La quatrième partie détaille le protocole de négociation proposé ainsi que la stratégie de décision adoptée.

## IV.2 Comportement de l'Agent Utilisateur dans un Voisinage Collaboratif

L'agent utilisateur $Au$ représente une interface entre l'utilisateur et le système. En effet, un agent $Au$ permet à l'utilisateur de se connecter au système, formuler ses requêtes, demander des services, saisir ses préférences, mettre à jour des données, afficher les résultats, etc.

Lorsque l'utilisateur se connecte au système, l'agent correspondant (i.e. $Au$) récupère la dernière mise à jour d'une table appelée *table d'état*. Cette table contient des informations de routages des clients actifs dans la coalition. La table d'état sera détaillée dans le chapitre IV.

Grâce à cette table, l'agent $Au$ peut identifier les fournisseurs des services demandés (i.e. clients actifs). Le tableau IV.1 illustre un exemple d'une table d'état à l'instant $t = 10$ d'un voisinage collaboratif $K_1^{10}$ rattaché à un connecteur $C_1$.

TABLE IV.1: Exemple d'une table d'état

| **Identifiant du service** | **Fournisseurs disponibles** |
|---|---|
| $ids_2$ | $\hat{a}_{3,1}$, $\hat{a}_{15,1}$, $\hat{a}_{5,1}$ |
| $ids_5$ | $\hat{a}_{4,1}$, $\hat{a}_{12,1}$ |
| $ids_{13}$ | $\hat{a}_{3,1}$, $\hat{a}_{4,1}$, $\hat{a}_{6,1}$, $\hat{a}_{9,1}$ |
| $ids_8$ | $\hat{a}_{15,1}$, $\hat{a}_{6,1}$, $\hat{a}_{5,1}$ |
| $ids_{10}$ | $\hat{a}_{9,1}$ |
| $ids_4$ | $\hat{a}_{4,1}$, $\hat{a}_{12,1}$ |
| $ids_9$ | $\hat{a}_{2,1}$, $\hat{a}_{3,1}$, $\hat{a}_{8,1}$ |

Une fois la requête est formulée par un agent utilisateur ($Au$) en utilisant la table d'état pour identifier les clients actifs disponibles dans le voisinage, la l'agent $Au$ décompose la requête en deux sous ensembles de services comme l'explique le paragraphe IV.4. Le premier sous ensemble est envoyé au agents actifs disponibles dans le voisinage. Le deuxième est envoyé au connecteur et plus spécifiquement à un agent $Ad$ disponible. Ce dernier est associé à une société d'agents préalablement créée et tenue en état d'attente. S'il n'existe encore pas d'agent $Ad$ disponible dans le connecteur, un nouvel agent $Ad$ est automatiquement créé au sein d'une nouvelle société d'agents. Dès que l'agent $Au$ reçoit une réponse donnée de la part de l'agent $Ac$, il affiche le résultat obtenu.

Si le résultat obtenu est accompagné d'une permission de changement de rôle, l'agent $Au$ le garde dans sa mémoire (i.e. la mémoire du terminal mobile) afin de l'offrir aux membre du voisinage collaboratif jusqu'à ce que la date de fin de validité de l'information correspondante à ce résultat est atteinte. Si l'agent $Au$ n'est pas autorisé à changer de rôle ou quand la validité de l'information arrive à échéance, le résultat est supprimé automatiquement de la mémoire du terminal mobile afin de libérer les ressources.

Une fois les services demandés acquis et si une permission d'avoir un rôle actif est délivrée, l'agent $Au$ change de rôle et s'inscrit dans la liste des fournisseurs de services au sein de sa coalition (i.e. la table d'état). Il peut ensuite entrer en interaction avec d'autres agents utilisateurs afin de leur fournir le(s) service(s) stocké(s) dans sa mémoire. Ces derniers peuvent être dynamiques. Dans ce cas, le stockage

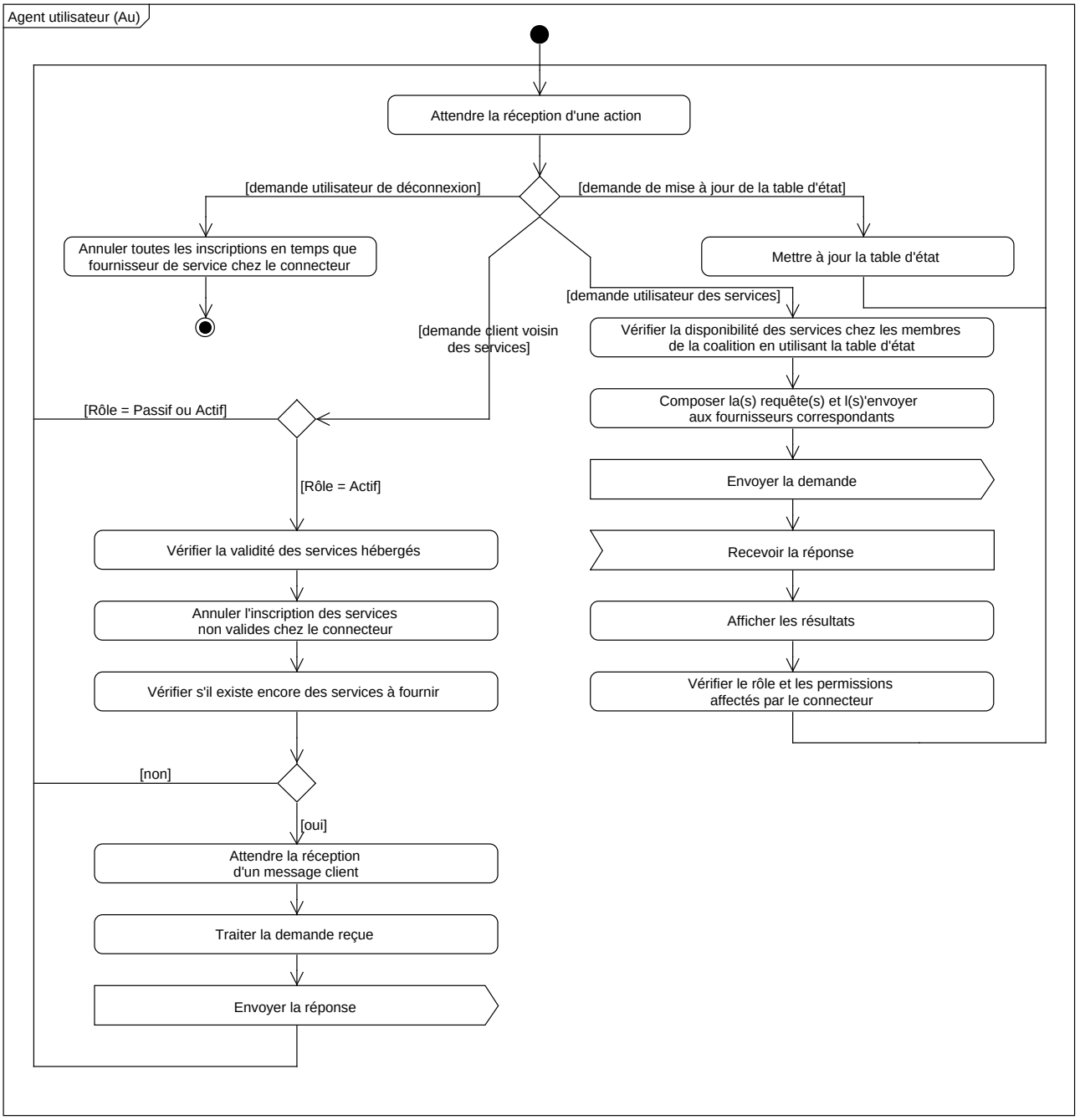



FIGURE IV.1: Diagramme d'activité d'un Agent utilisateur ($Au$).

est limité à une période de temps qui correspond à la durée de validité de l'information. La Fig. IV.1 présente le diagramme d'activité du comportement d'un agent utilisateur.

Le paragraphe suivant présente en détail la stratégie de changement de rôle des agents utilisateurs au sein d'un voisinage collaboratif. Il introduit en particulier les notions de *rareté* et de *popularité* sur lesquelles repose cette stratégie.

## IV.3 Administration du Changement Dynamique du Rôle

L'objectif principal du processus d'administration du changement dynamique du rôle est l'amélioration des performances du système. Nous nous focalisons sur la minimisation des temps de réponse aux requêtes utilisateurs tout en gardant la capacité du système à supporter un nombre élevé de requêtes simultanées. Pour cela, deux indices clés pour l'administration du changement dynamique du rôle sont identifiés : l'indice de *rareté* et l'indice de *popularité*. Ces deux indices permettent au système de décider quand et quel client est a la permission de changer de rôle et devenir actif. Nous définissons la signification de ces deux indices dans le paragraphe suivant.

### IV.3.1 Indices de Gestion des Permissions

Avant de définir les indices de rareté et de popularité, nous présentons un exemple afin d'illustrer le sens de chaque indice.

**Exemple.** Pendant la période d'hiver, certains médicaments sont fréquemment demandés aux pharmacies. Pour cela, le pharmacien a besoin de bien gérer son stock pour ne pas être en situation de rupture de stock qui peut gêner aussi bien l'acheteur que le vendeur. Le pharmacien peut gérer son stock selon plusieurs méthodes. En effet, il peut se réapprovisionner d'une manière constante, c'est-à-dire, demander des quantités fixes à des dates fixes. Cependant, les médicaments qui sont très demandés pendant la période d'hiver, ne le sont pas forcément pendant la période de l'été par exemple. Par conséquent, le mieux à faire est de commander des quantités variables à des dates variables, c'est-à-dire, à la demande.

D'après cet exemple, l'indice de rareté d'un médicament n'est autre que sa situation actuelle correspondant à sa quantité en stock. Quant à l'indice de popularité, ce n'est d'autre que l'estimation de la demande de ce médicament pendant une période de temps bien déterminée. Dans ce cas, nous proposons les définitions suivantes :

**Définition 6.** ***Rareté*** : la rareté d'un service peut être définie comme étant le niveau de redondance de ce service au sein d'un voisinage collaboratif. Autrement dit, c'est le nombre d'agents actifs qui proposent ce service au sein de la coalition. La rareté (ou niveau de redondance) d'un service $ids_m$ dans un voisinage collaboratif $K_j^t$ est notée par $RL_{m,j}^t$.

**Définition 7.** ***Popularité*** : la popularité d'un service peut être définie comme étant l'estimation de la demande de ce service pendant une durée de temps bien déterminée. Cette estimation peut être en

fonction du temps et de l'espace puisque la demande des services peut être elle aussi une fonction du temps et de l'espace.

Une estimation convenable pour chaque service peut être fixée en se basant sur des enquêtes de satisfaction et de préférences. Mathématiquement parlant, l'estimation de la demande peut être modélisée par un seuil minimal de redondance de ce service au sein du voisinage collaboratif. La popularité d'un service $ids_m$ proposé par un connecteur $C_j$ est notée par $popularite_{m,j}$.

**Définition 8.** ***Permission*** : la permission est l'accord délivré par le connecteur à un agent utilisateur pour changer son rôle en client actif. Cette permission détermine quel(s) service(s) cet agent va fournir et pour quelle(s) durée(s) de temps.

Les indices de rareté et de popularité sont introduits afin de répondre aux questions suivantes :
– Quand un service est rare ?
– Quand un agent utilisateur est autorisé à changer de rôle ?
– Est-ce qu'un niveau de redondance est atteint ou pas ?
Pour répondre à ces questions, nous proposons tout d'abord de définir quelques variables que nous regroupons dans la table IV.2.

TABLE IV.2: Récapitulatif des variables du système

| **Notation** | **Déscription** |
|---|---|
| $\widehat{NA}^t_{m,j}$ | nombre d'agents actifs fournisseurs du service $ids_m$ liés à un connecteur $C_j$ à l'instant $t$ |
| $popularite_{m,j}$ | niveau de *popularité* d'un service $ids_m$ dans un connecteur $C_j$ |
| $RL^t_{m,j}$ | *niveau de redondance* (rareté) d'un service $ids_m$ dans un connecteur $C_j$ à l'instant $t$ |
| $DV_m$ | date de fin de validité d'un service $ids_m$ |
| $ST^t_j$ | *table d'état* d'un voisinage collaboratif $K^t_j$ |

Rappelons que $\hat{N}^t_j$ (resp. $N^t_j$) est le nombre de clients actifs (resp. clients passifs) dans un voisinage collaboratif $K^t_j$ à l'instant $t$. $RL^t_{m,j}$ est donné par la formule suivante :

$$RL^t_{m,j} = \frac{\widehat{NA}^t_{m,j}}{\hat{N}^t_j + N^t_j} \tag{IV.1}$$

### IV.3.2 Algorithme d'Administration du Rôle Dynamique

Étant donné que les agents utilisateurs se connectent et se déconnectent du système d'une façon dynamique et aléatoire, le rôle de l'algorithme d'administration du changement dynamique du rôle

(Algorithme 7) est de vérifier continuellement le niveau de redondance des services dans le voisinage collaboratif. En effet, les utilisateurs qui demandent des services ne sont pas tous autorisés à changer de rôle et devenir actifs. A chaque fois qu'un utilisateur se connecte et demande des services ou se déconnecte (et se désinscrit de la liste des fournisseurs de service s'il est actif), le niveau de redondance est mis à jour.

Dans ce cas, l'objectif de l'algorithme est de n'attribuer les droits à changer de rôle que si le niveau de redondance de chaque service au dessous de la valeur de popularité correspondante. En effet, si le connecteur attribue à tous les agents utilisateurs (tous les membres d'une coalition) des permissions pour changer leurs rôles (passer d'un rôle passif à un rôle actif), nous pouvons garantir qu'il existe suffisamment de fournisseurs de services au sein d'un voisinage collaboratif. Néanmoins, cette pratique entraîne l'inondation du réseau par des messages de communication lors de la recherche des services. En effet, chaque demande de services génère un tour de négociation qui se déroule entre un initiateur (l'agent demandeur) et un ensemble de participants (les agents actifs proposant le(s) service(s) demandé(s)) (§IV.6). Pour cela, nous avons développé cette stratégie basée sur le niveau de redondance qui vise à garder un nombre minimal de fournisseurs de services dans un voisinage collaboratif tout en minimisant le flux de communication dans le réseau. Le choix de cette stratégie est appuyé par les résultats de simulation présentés dans le prochain chapitre.

La table d'état $ST_j^t$ d'un voisinage collaboratif (ou coalition) est une table de hachage qui permet d'associer des clés à des valeurs. Les clés dans notre cas ne sont autres que les identifiants des services, quant aux valeurs associées à chaque clé ce sont les agents actifs proposant le service correspondant. La table IV.3 présente un exemple d'une table d'état $ST_2^{100}$ d'une coalition $K_2^{100}$ à l'instant $t = 100$.

TABLE IV.3: Exemple d'une table d'état $ST_2^{100}$

| **Identifiant de service** | **Fournisseurs de service disponibles** |
|---|---|
| $ids_1$ | $\hat{a}_{1,2}, \hat{a}_{2,2}, \hat{a}_{6,2}$ |
| $ids_2$ | $\hat{a}_{3,2}, \hat{a}_{12,2}, \hat{a}_{7,2}, \hat{a}_{8,2}, \hat{a}_{11,2}, \hat{a}_{5,2}, \hat{a}_{1,2}$ |
| $ids_3$ | $\hat{a}_{2,2}, \hat{a}_{8,2}, \hat{a}_{19,2}, \hat{a}_{31,2}, \hat{a}_{5,2}, \hat{a}_{17,2}, \hat{a}_{6,2}, \hat{a}_{7,2}$ |
| $ids_4$ | $\hat{a}_{10,2}, \hat{a}_{13,2}, \hat{a}_{21,2}, \hat{a}_{2,2}, \hat{a}_{15,2}, \hat{a}_{4,2}, \hat{a}_{9,2}, \hat{a}_{17,2}, \hat{a}_{5,2}, \hat{a}_{6,2}$ |
| $ids_5$ | $\hat{a}_{2,2}, \hat{a}_{7,2}$ |

Dans le cadre de cette thèse, nous proposons les hypothèses suivantes :

- les informations reçues par un utilisateur suite à la demande d'un ensemble de services auprès du connecteur ou des clients actifs du voisinage collaboratif sont gardées dans la mémoire de son terminal mobile tant que ces informations sont pertinentes ;
- à l'issu de la date de validité d'un service, ce dernier est supprimé automatiquement de la mémoire du terminal ;

– La problématique de gestion de la mémoire d'un terminal mobile n'est pas traitée dans cette thèse.

**Algorithme 7** Algorithme d'administration du changement dynamique du rôle

**Entrée:** $ST_j^t$, $N_j^t$, $\hat{N}_j^t$
**Sortie:** Attribution dynamique des permissions
1: **si** une requête est reçue d'un agent $a_{h,j}$ **alors**
2: $\forall ids_m$ demandé
3: $\widehat{NA}_{m,j}^t \leftarrow$ nombre de fournisseurs de $ids_m$ dans $ST_j^t$
4: $RL_{m,j}^t \leftarrow (\widehat{NA}_{m,j}^t / (\hat{N}_j^t + N_j^t))$
5: **si** $RL_{m,j}^t < popularite_{m,j}$ **alors**
6: Envoyer une permission à l'agent $a_{h,j}$ pour l'autoriser à fournir $ids_m$ jusqu'à $DV_m$
7: **fin si**
8: **fin si**

Dans ce qui suit, nous présentons un exemple d'application de l'algorithme 7 responsable de l'administration du changement dynamique des rôles des agents utilisateurs.

### IV.3.3 Exemple d'Application

Dans cet exemple, nous considérons un connecteur $C_1$ proposant 5 services $\{ids_1, ids_2, ids_3, ids_4, ids_5\}$ et un voisinage collaboratif $K_1^t$ composé de 6 agents utilisateurs $\{a_{1,1}, a_{2,1}, a_{3,1}, a_{4,1}, a_{5,1}, a_{6,1}\}$. Les indices de popularité de chaque service sont illustrés dans la table IV.4.

TABLE IV.4: Exemple d'indices de popularité d'un ensemble de services

| **Service** | $ids_1$ | $ids_2$ | $ids_3$ | $ids_4$ | $ids_5$ |
|---|---|---|---|---|---|
| **Popularité** | 0,21 | 0,05 | 0,1 | 0,07 | 0,18 |

Le scénario de l'exemple se déroule comme suit : les agents utilisateurs se connectent au système l'un après l'autre (afin de mettre l'accent sur le processus de gestion des permissions) et demandent à chaque fois un ensemble de services parmi les services proposés par le système (connecteur et voisinage collaboratif). La Fig. IV.2 illustre une représentation schématique de l'évolution dynamique du voisinage collaboratif.

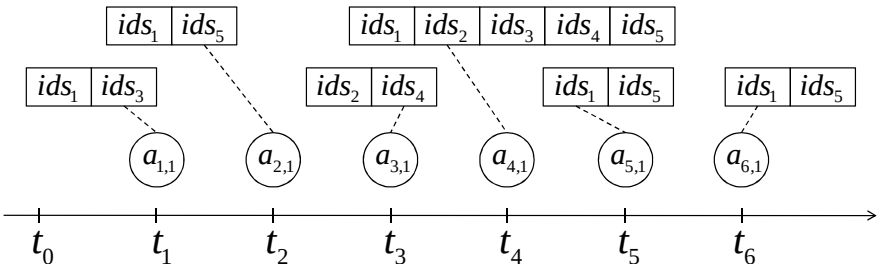



FIGURE IV.2: Exemple d'évolution d'une coalition $K_1^t$ composée de 6 agents utilisateurs.

L'affectation dynamique des permissions de cet exemple se déroule comme le présente la table IV.5. L'algorithme dans ce cas se comporte conformément à l'objectif principal puisque les services qui possèdent des popularités élevées sont plus susceptibles d'être fournis par les clients actifs.

TABLE IV.5: Exemple de scénario d'affectation dynamique des permissions

| $t$ | **Identifiant du service** | **Historique des requêtes** | $\widehat{NA}^t_{m,j}$ | $RL^t_{m,j}$ | **Permissions** |
|---|---|---|---|---|---|
| $t_0$ | $ids_1$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_2$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_3$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_4$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_5$ | $\varnothing$ | 0 | 0 | aucune |
| $t_1$ | $ids_1$ | $a_{1,1}$ | 0 | 0 | $a_{1,1}$ |
| | $ids_2$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_3$ | $a_{1,1}$ | 0 | 0 | $a_{1,1}$ |
| | $ids_4$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_5$ | $\varnothing$ | 0 | 0 | aucune |
| $t_2$ | $ids_1$ | $a_{1,1}/a_{2,1}$ | 1 | 0.5 | aucune |
| | $ids_2$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_3$ | $a_{1,1}$ | 1 | 0.5 | aucune |
| | $ids_4$ | $\varnothing$ | 0 | 0 | aucune |
| | $ids_5$ | $a_{2,1}$ | 0 | 0 | $a_{2,1}$ |
| $t_3$ | $ids_1$ | $a_{1,1}/a_{2,1}$ | 1 | 0.33 | aucune |
| | $ids_2$ | $a_{3,1}$ | 0 | 0 | $a_{3,1}$ |
| | $ids_3$ | $a_{1,1}$ | 1 | 0.33 | aucune |
| | $ids_4$ | $a_{3,1}$ | 0 | 0 | $a_{3,1}$ |
| | $ids_5$ | $a_{2,1}$ | 1 | 0.33 | aucune |
| $t_4$ | $ids_1$ | $a_{1,1}/a_{2,1}/a_{4,1}$ | 1 | 0.25 | aucune |
| | $ids_2$ | $a_{3,1}/a_{4,1}$ | 1 | 0.25 | aucune |
| | $ids_3$ | $a_{1,1}/a_{4,1}$ | 1 | 0.25 | aucune |
| | $ids_4$ | $a_{3,1}/a_{4,1}$ | 1 | 0.25 | aucune |
| | $ids_5$ | $a_{2,1}/a_{4,1}$ | 1 | 0.25 | aucune |
| $t_5$ | $ids_1$ | $a_{1,1}/a_{2,1}/a_{4,1}/a_{5,1}$ | 1 | 0.2 | $a_{5,1}$ |
| | $ids_2$ | $a_{3,1}/a_{4,1}$ | 1 | 0.2 | aucune |
| | $ids_3$ | $a_{1,1}/a_{4,1}$ | 1 | 0.2 | aucune |
| | $ids_4$ | $a_{3,1}/a_{4,1}$ | 1 | 0.2 | aucune |
| | $ids_5$ | $a_{2,1}/a_{4,1}/a_{5,1}$ | 1 | 0.2 | aucune |
| $t_6$ | $ids_1$ | $a_{1,1}/a_{2,1}/a_{4,1}/a_{5,1}$ | 2 | 0.33 | aucune |
| | $ids_2$ | $a_{3,1}/a_{4,1}$ | 1 | 0.17 | aucune |
| | $ids_3$ | $a_{1,1}/a_{4,1}/a_{6,1}$ | 1 | 0.17 | aucune |
| | $ids_4$ | $a_{3,1}/a_{4,1}$ | 1 | 0.17 | aucune |
| | $ids_5$ | $a_{2,1}/a_{4,1}/a_{5,1}/a_{6,1}$ | 1 | 0.17 | $a_{6,1}$ |

Comme nous pouvons voir dans la table IV.5, à chaque fois qu'un service est demandé, l'algorithme d'administration du changement de rôle dynamique calcule le niveau de redondance associé à ce service et délivre les permissions nécessaires pour avoir un bon niveau de balance des charges dans la coalition. Il faut remarquer que quand l'agent utilisateur $a_{4,1}$ rejoint dans la coalition à $t = t_4$ et

compose sa requête $req_{4,1} = \{ids_1, ids_2, ids_3, ids_4, ids_5\}$, le système n'accorde aucune permission à cet agent puisque le niveau de redondance des 5 services demandés est supérieur aux seuils de redondance correspondants. Finalement, la table d'état à l'instant $t_6$ est illustrée dans la table IV.6.

TABLE IV.6: Exemple d'une table d'état $ST_1^6$ à l'instant $t = t_6$

| **Identifiant de service** | **Clients actifs disponibles** |
|---|---|
| $ids_1$ | $\hat{a}_{1,1}, \hat{a}_{5,1}$ |
| $ids_2$ | $\hat{a}_{3,1}$ |
| $ids_3$ | $\hat{a}_{1,1}$ |
| $ids_4$ | $\hat{a}_{3,1}$ |
| $ids_5$ | $\hat{a}_{2,1}, \hat{a}_{6,1}$ |

Nous rappelons que dans le diagramme d'activité d'un agent utilisateur, la récupération de la table d'état est parmi les toutes premières tâches effectuées lors de la connexion au système. En effet, cette table permet à l'agent $Au$ de rechercher et découvrir les fournisseurs de services (i.e. les clients actifs) dans son voisinage. Les processus de recherche de services et d'exécution des requêtes utilisateurs par l'agent $Au$ sont présentés dans le paragraphe suivant.

## IV.4 Recherche des IMA dans un Voisinage Collaboratif

La recherche des IMA au sein d'un voisinage collaboratif est une tâche déléguée aux agents utilisateurs. En effet, avant d'envoyer une requête composée de services demandés par l'utilisateur au connecteur, l'agent $Au$ effectue une recherche locale, c'est-à-dire au sein de la coalition, afin de vérifier si les services (ou une partie de ces services) sont déjà disponibles chez d'autres agents $Au$ qui les proposent.

Par conséquent, afin d'optimiser les communications entre les clients (i.e. les agents) du système, ces derniers vont être amenés à sélectionner leurs voisins (i.e. les membres d'un même voisinage collaboratif) qui possèdent le taux de transfert des données le plus élevé. Les agents demandeurs d'information doivent considérer l'efficacité de transfert des informations lorsqu'ils choisissent leurs fournisseurs de services. Pour des raisons de simplification, nous supposons que les agents sont situés dans un espace bidimensionnel où les distances linéaires qui les séparent sont inversement proportionnelles aux taux de transfert.

Quand la distance entre deux clients augmente, la vitesse de transfert des données entre les deux diminue. Dans ce cas, notre objectif est d'aider les agents utilisateurs à demander les services à leurs voisins les plus proches afin de garantir une communication efficace et réduire les délais de transmission. Par conséquent, les tâches de découverte et de sélection des fournisseurs de service (i.e. clients actifs)

d'un même voisinage collaboratif se basent sur leurs positions par rapport à l'agent demandeur de service. Pour chaque membre du voisinage collaboratif nous définissons un *rayon du voisinage* :

**Définition 9. Rayon du voisinage** : le rayon d'un voisinage $K_j^t$, noté par $fmax_j$, est le nombre maximal d'agents actifs avec lesquels un demandeur de service peut interagir. Ce nombre est généré en fonction du nombre total des clients connectés et du flux de communication sur le réseau.

### IV.4.1 Algorithme d'Identification des Fournisseurs de Services dans un Voisinage Collaboratif

En utilisant les tables d'états, les agents utilisateurs sont capables de localiser les services demandés d'une manière autonome. Ainsi, en utilisant l'algorithme 8, les clients peuvent identifier deux ensembles de services :

- $PK_{h,j}$ : l'ensemble des identifiants des services qui peuvent être fournis par $\hat{A}_j^t$ ;
- $PC_{h,j}$ : l'ensemble des identifiants des services qui ne peuvent être fournis que par $C_j$ au moment du processus d'identification. Sachant que $PK_{h,j} \cap PC_{h,j} = \varnothing$ et que $PK_{h,j} \cup PC_{h,j} = req_{h,j}$.

**Algorithme 8** Identification des fournisseurs de services dans une coalition pour un utilisateur

**Entrée:** $req_{h,j}$,$\hat{A}_j^t$, $ST_j^t$, $fmax_j$
**Sortie:** Table de fournisseurs de services $CT_{h,j}^t$

Initialiser une liste $PK_{h,j}$
Initialiser une liste $PC_{h,j}$
Initialiser une table de hachage $CT_{h,j}^t$
Initialiser un compteur $k \leftarrow 1$
Initialiser un compteur $l \leftarrow 1$
//Étape 1 : identification de $PK_{h,j}$ et $PC_{h,j}$
**pour** $i = 1$ jusqu'à $Cardinal(req_{h,j})$ **faire**
  **si** $ST_j^t[req_{h,j}[i]]$ est une liste non vide **alors**
    $PK_{h,j}[k] \leftarrow req_{h,j}[i]$
    $k \leftarrow k + 1$
  **sinon**
    $PC_{h,j}[l] \leftarrow req_{h,j}[i]$
    $l \leftarrow l + 1$
  **fin si**
**fin pour**
//Étape 2 : sélection des fournisseurs de $req_{h,j}$
Récupérer les positions des fournisseurs de $PK_{h,j}$
**pour** $i = 1$ jusqu'à $Cardinal(PK_{h,j})$ **faire**
  Initialiser une liste $L \leftarrow$ fournisseurs de $PK_{h,j}[i]$ dans $ST_j^t[PK_{h,j}[i]]$
  Ajouter les $fmax_j$ fournisseurs les plus proches à $CT_{h,j}^t[PK_{h,j}[i]]$
**fin pour**
**retourner** $CT_{h,j}^t$

L'algorithme d'identification récupère la requête $req_{h,j}$ de l'utilisateur composée de l'ensemble des identifiants des services demandés. Ensuite, en utilisant la table d'état $ST_j^t$, il procède à une identification itérative afin de créer deux sous-requêtes ; la sous-requête demandant les services fournis par les membres du voisinage collaboratif et la sous-requête demandant les services non disponibles dans le voisinage collaboratif noté par $PK_{h,j}$ et par $PC_{h,j}$.

Une fois les fournisseurs de services identifiés, l'algorithme effectue la sélection des $fmax_j$ fournisseurs de services les plus proches afin de passer à l'étape de négociation que nous détaillerons dans le paragraphe IV.6. Dans ce qui suit, nous présentons un exemple d'application de l'algorithme d'identification des fournisseurs de services dans un voisinage collaboratif.

### IV.4.2 Exemple d'Application

Dans cet exemple, nous considérons :

- un connecteur $C_1$ proposant un ensemble de 30 services $\{ids_1, ids_2, .., ids_{30}\}$ ;
- un voisinage collaboratif composé de 38 agents utilisateurs $\{a_{1,1}, a_{2,1}, .., a_{38,1}\}$ ;

Nous supposons que tous les services sont disponibles et peuvent être fournis par les clients actifs du voisinage collaboratif. Nous considérons aussi la répartition illustrée dans la Fig. IV.3, de l'ensemble des clients (l'annexe C présente une table regroupant les coordonnées de chaque client de l'exemple).

Nous considérons aussi la table d'état $ST_1^{100}$ (table IV.7) associée à la coalition $K_1^{100}$ à l'instant $t = 100$. Supposons que l'agent $a_{7,1}$ compose la requête $req_{7,1} = \{ids_4, ids_5, ids_6, ids_{10}, ids_{11}, ids_{13}\}$ à l'instant $t = 100$.

En utilisant la table d'état $ST_1^{100}$, l'algorithme d'identification des fournisseurs de services génère tout d'abord les deux sous-ensembles suivants :

- $PK_{7,1} = \{ids_4, ids_5, ids_6, ids_{10}, ids_{11}, ids_{13}\} = req_{7,1}$ ;
- $PC_{7,1} = \varnothing$.

L'ensemble $PC_{7,1}$ est vide puisque tous les services demandés sont disponibles dans la coalition à l'instant $t = 100$. Ensuite, l'algorithme sélectionne les fournisseurs les plus proches et crée la table de fournisseurs de services $CT_{7,1}^{100}$ , illustrée dans le tableau IV.8, avec un rayon de voisinage $fmax_1 = 3$.

La table de fournisseurs des services permet à l'agent utilisateur d'identifier les clients actifs avec lesquels il établira la communication afin de récupérer les services demandés suite à une négociation. Le processus de négociation est présenté dans la partie suivante.

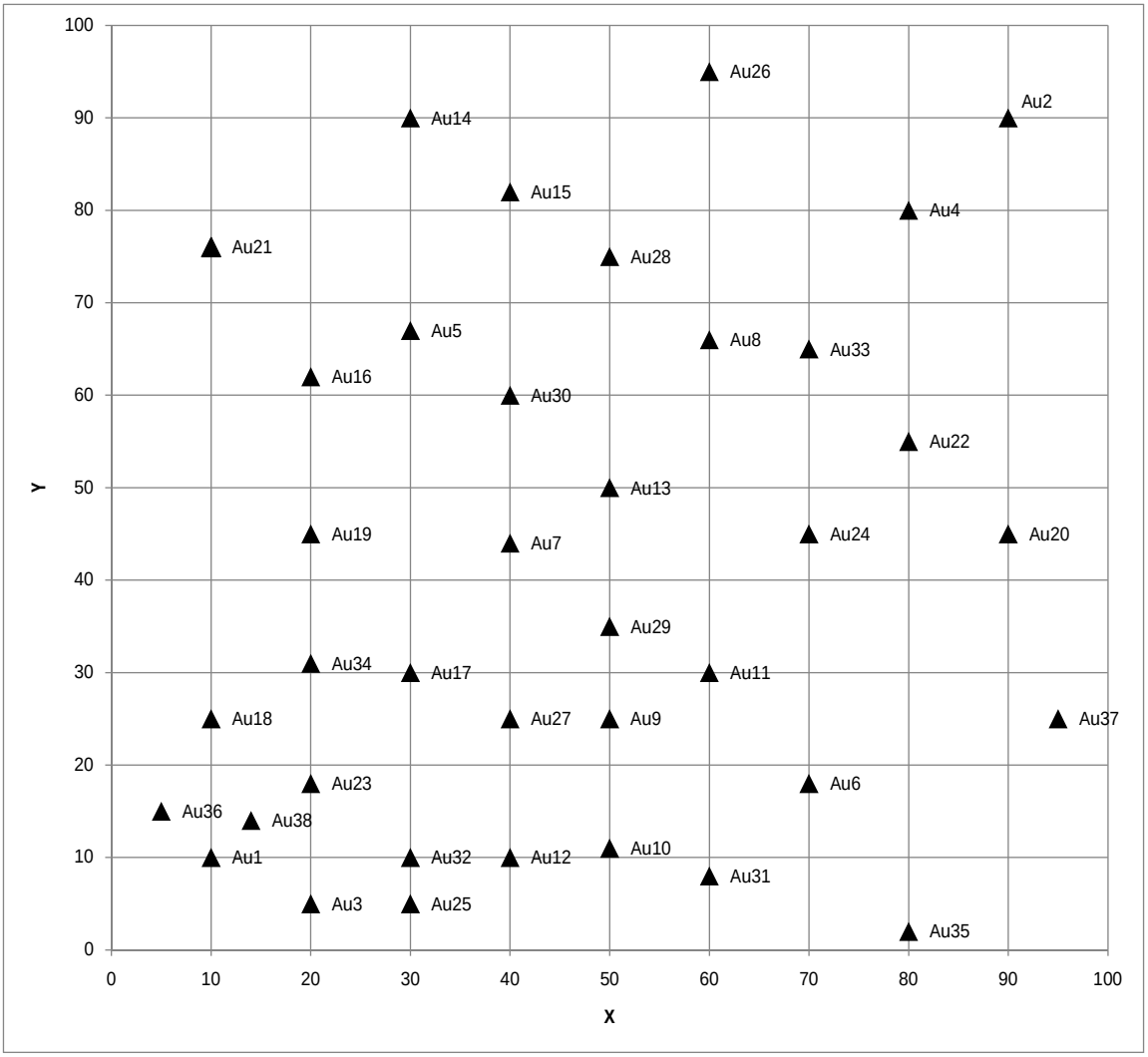



FIGURE IV.3: Exemple de répartition aléatoire des agents $Au$ dans un espace ubiquitaire

## IV.5 Protocoles et Ontologies dans les Systèmes Multi-Agents

Un protocole est défini comme étant un ensemble de règles que les entités d'un système utilisent pour communiquer lors d'un échange d'informations. Le système proposé dans cette thèse (Chapitre III) intègre un protocole de communication innovant basé sur échange à accord partiel. Dans ce qui suit, les spécifications de quelques protocoles de communication dans les SMA sont présentées. La notion d'ontologie dans les SMA est aussi présentée.

TABLE IV.7: Table d'état $ST_1^{100}$ à l'instant $t = 100$

| **Identifiant de service** | **Fournisseurs de services disponibles** |
|---|---|
| $ids_1$ | $\hat{a}_{1,1}$, $\hat{a}_{2,1}$, $\hat{a}_{8,1}$ |
| $ids_2$ | $\hat{a}_{1,1}$, $\hat{a}_{8,1}$ |
| $ids_3$ | $\hat{a}_{9,1}$, $\hat{a}_{10,1}$ |
| $ids_4$ | $\hat{a}_{10,1}$, $\hat{a}_{12,1}$, $\hat{a}_{16,1}$ |
| $ids_5$ | $\hat{a}_{15,1}$, $\hat{a}_{16,1}$, $\hat{a}_{17,1}$, $\hat{a}_{5,1}$, $\hat{a}_{1,1}$, $\hat{a}_{8,1}$, $\hat{a}_{33,1}$ |
| $ids_6$ | $\hat{a}_{12,1}$, $\hat{a}_{2,1}$, $\hat{a}_{3,1}$, $\hat{a}_{4,1}$, $\hat{a}_{5,1}$, $\hat{a}_{6,1}$ |
| $ids_7$ | $\hat{a}_{32,1}$, $\hat{a}_{16,1}$, $\hat{a}_{19,1}$, $\hat{a}_{23,1}$, $\hat{a}_{24,1}$, $\hat{a}_{25,1}$ |
| $ids_8$ | $\hat{a}_{8,1}$, $\hat{a}_{6,1}$, $\hat{a}_{10,1}$, $\hat{a}_{1,1}$, $\hat{a}_{20,1}$, $\hat{a}_{30,1}$ |
| $ids_9$ | $\hat{a}_{4,1}$, $\hat{a}_{18,1}$, $\hat{a}_{22,1}$, $\hat{a}_{24,1}$, $\hat{a}_{26,1}$ |
| $ids_{10}$ | $\hat{a}_{6,1}$, $\hat{a}_{17,1}$, $\hat{a}_{8,1}$, $\hat{a}_{13,1}$, $\hat{a}_{14,1}$ |
| $ids_{11}$ | $\hat{a}_{17,1}$, $\hat{a}_{25,1}$, $\hat{a}_{26,1}$, $\hat{a}_{29,1}$ |
| $ids_{12}$ | $\hat{a}_{38,1}$, $\hat{a}_{30,1}$, $\hat{a}_{29,1}$, $\hat{a}_{1,1}$, $\hat{a}_{2,1}$ |
| $ids_{13}$ | $\hat{a}_{32,1}$, $\hat{a}_{33,1}$, $\hat{a}_{34,1}$, $\hat{a}_{35,1}$ |
| $ids_{14}$ | $\hat{a}_{36,1}$, $\hat{a}_{37,1}$, $\hat{a}_{38,1}$ |
| $ids_{15}$ | $\hat{a}_{4,1}$, $\hat{a}_{5,1}$, $\hat{a}_{7,1}$ |

TABLE IV.8: Exemple d'une table de fournisseurs de services $CT_{7,1}^{100}$ à l'instant $t = 100$

| **Identifiant de service** | **Fournisseurs de services disponibles** |
|---|---|
| $ids_4$ | $\hat{a}_{16,1}$, $\hat{a}_{12,1}$, $\hat{a}_{10,1}$ |
| $ids_5$ | $\hat{a}_{17,1}$, $\hat{a}_{5,1}$, $\hat{a}_{16,1}$ |
| $ids_6$ | $\hat{a}_{5,1}$, $\hat{a}_{12,1}$, $\hat{a}_{6,1}$ |
| $ids_{10}$ | $\hat{a}_{13,1}$, $\hat{a}_{17,1}$, $\hat{a}_{8,1}$ |
| $ids_{11}$ | $\hat{a}_{29,1}$, $\hat{a}_{17,1}$, $\hat{a}_{25,1}$ |
| $ids_{13}$ | $\hat{a}_{34,1}$, $\hat{a}_{32,1}$, $a_{33,1}$ |

### IV.5.1 Protocoles de Négociation dans les SMA

#### IV.5.1.1 Le Protocole Contract Net Protocol (CNET)

CNET [Smith, 1980] est un protocole de haut niveau (couche application du modèle OSI [1]) qui permet d'établir une coopération efficace entre des nœuds communicants dans un réseau distribué de résolution des problèmes. Chaque unité de résolution peut avoir deux rôles : gestionnaire ou entrepreneur. Le nœud du gestionnaire est responsable de la supervision de l'exécution des tâches et du traitement des résultats fournis alors que le nœud de l'entrepreneur est celui qui exécute ces tâches.

Dans CNET, un nœud peut être un gestionnaire et un entrepreneur en même temps. Dans ce cadre, CNET est donc similaire à notre système puisque les clients peuvent changer de rôle d'une manière dynamique pour devenir des fournisseurs de services tout en restant des clients normaux (des demandeurs des services). Dans CNET, quand un nœud (gestionnaire) crée une tâche, il envoie un message d'*annonce* de tâche. Le message peut être diffusé à tous les nœuds disponibles (entrepreneurs) lorsque

1. Open Systems Interconnection

le gestionnaire ne dispose pas d'information sur les entrepreneurs disponibles ou intéressés par cette tâche. Sinon, le gestionnaire peut diffuser son message en unicast en s'adressant directement à l'entrepreneur concerné. Une fois que l'annonce de tâche est reçue par un entrepreneur, elle est traitée et une offre est produite en respectant les critères définis dans le message d'annonce de tâche (les spécifications d'admissibilité).

Le nœud du gestionnaire peut recevoir de nombreuses offres pour la même tâche, dans ce cas, il choisit la meilleure offre en se basant sur les informations contenues dans les messages de soumission d'offre. Plusieurs "tours" de négociation peuvent avoir lieu pour atteindre un compromis et dans ce cas un message *prix* est adressé au soumissionnaire correspondant et le processus de négociation se termine. Le manager peut envoyer un message au gestionnaire pour demander des informations par rapport au résultat d'exécution de la tâche. Dans ce cas, le gestionnaire doit renvoyer les informations nécessaires. La Fig. IV.4 illustre le diagramme de séquence du protocole CNET. Ce protocole de négociation permet de gérer et de contrôler les nœuds dans les systèmes d'information distribués. Cependant, il possède quelques faiblesses.

Tout d'abord, CNET suppose implicitement que chaque tâche annoncée possède au moins un nœud capable de l'exécuter ce qui ne peut pas être toujours vrai dans un système distribué. Ensuite, CNET ne spécifie pas une règle particulière pour le cas où les nœuds du système sont dynamiques, dans le sens où les nœuds du réseau se connectent et se déconnectent du système d'une manière dynamique. CNET suppose aussi qu'un nœud entrepreneur possède la liberté d'effectuer ou pas des offres pour les tâches annoncées. Dans ce cas, aucune garantie n'est offerte au gestionnaire pour trouver avec certitude un entrepreneur convenable capable de résoudre la tâche annoncée. Dans le cas où un contrat est modifié en raison du temps d'expiration et un nouveau contrat est annoncé, la situation reste toujours la même et peut provoquer une boucle infinie de tours de négociation.

#### IV.5.1.2 Le Protocole ANTS

ANTS [Mathieu et Verrons, 2003] est très similaire à CNET car il s'agit d'un protocole de négociation basé sur un contrat et a pratiquement les mêmes types de messages utilisés pour soumettre des annonces et recevoir des offres. Le processus de négociation se produit entre un initiateur et un ensemble de participants. Il commence par une proposition de contrat (annonce de la tâche dans CNET). Après avoir reçu un contrat, les participants peuvent l'accepter ou le refuser. Si le nombre minimal d'accords est atteint, l'initiateur confirme le contrat. Dans ce cas, la négociation réussit, puis se termine.

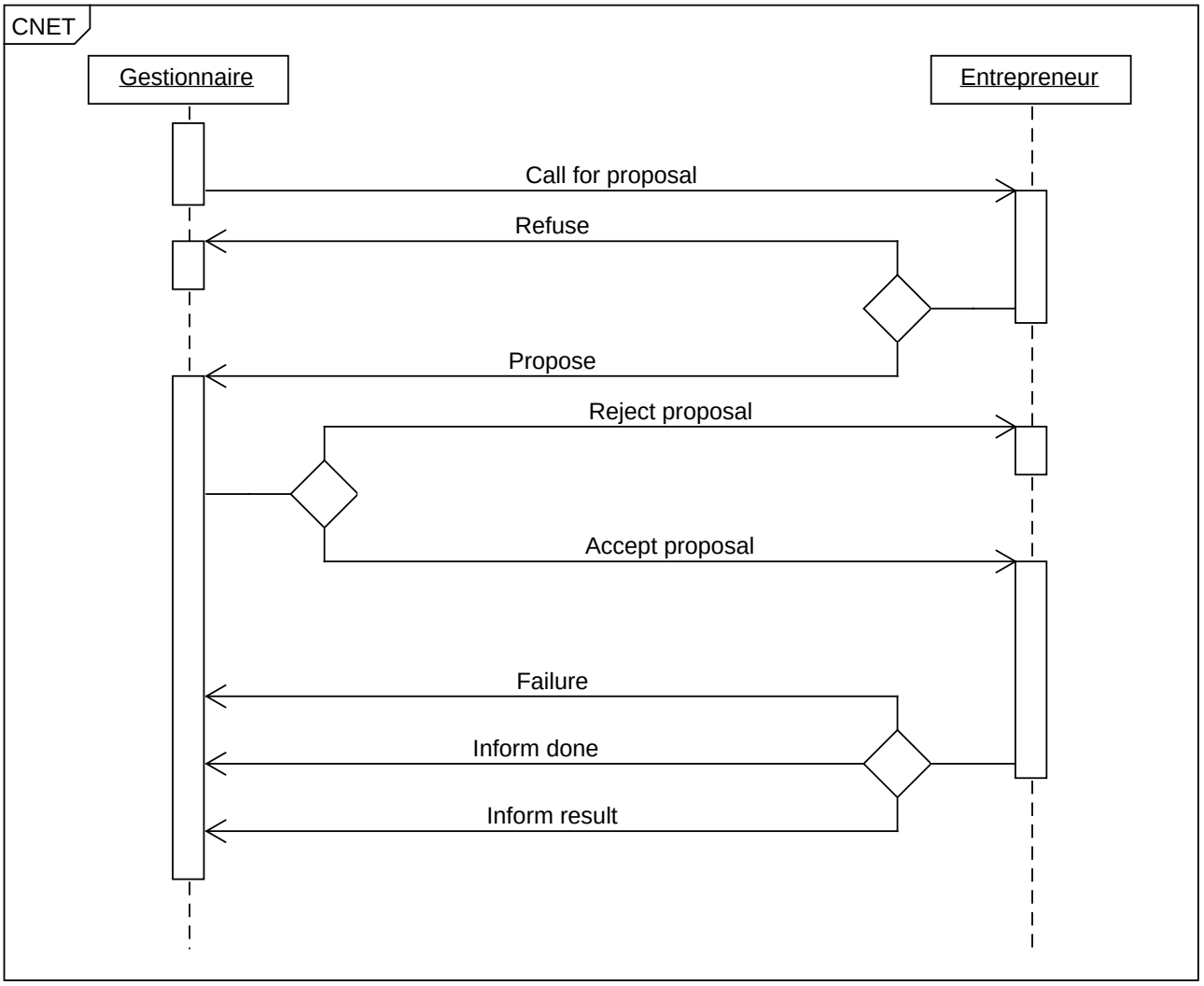



FIGURE IV.4: Illustration du diagramme de séquence du protocole CNET.

Dans le cas contraire, l'initiateur envoie une demande de modification (équivalent à une nouvelle annonce de tâche dans CNET) pour inviter les participants à renvoyer un contrat modifié qui reflète leurs préférences. Le processus s'enchaîne jusqu'à ce qu'un accord minimal soit atteint ou que le contrat soit annulé. ANTS cherche le nombre minimal d'accords alors que CNET essaye de trouver la meilleure offre. ANTS propose une technique de supervision des contrats basé sur des objectifs (créés par des initiateurs) et des engagements (créés par les participants) pour permettre plusieurs négociations simultanées. Dans ANTS, de façon similaire à CNET, un nœud peut être un initiateur et un participant en même temps. La Fig. IV.5 illustre le diagramme de séquence du protocole ANTS.

Afin d'éviter des situations d'impasse, des objectifs et des engagements sont classés dans une matrice de gestion des ressources qui permet à un contrat d'être négocié seulement lorsque toutes les ressources nécessaires sont libres. Le protocole ANTS ne fournit pas aux initiateurs l'assurance de trouver avec

certitude un nombre minimal d'accords afin de réussir le processus de négociation. Dans ce cas, une boucle infinie de négociation peut se produire.

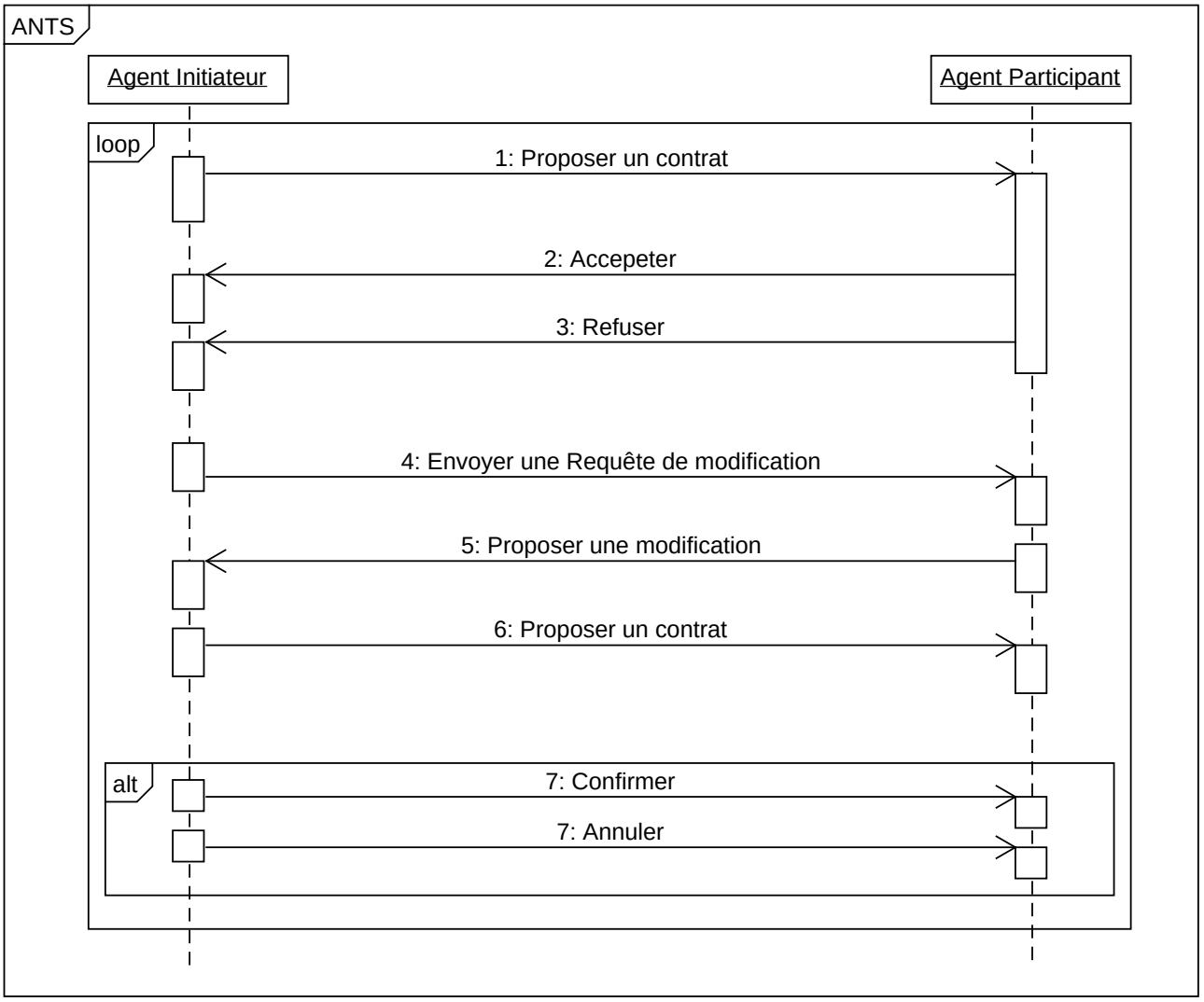



FIGURE IV.5: Illustration du diagramme de séquence du protocole ANTS.

#### IV.5.1.3 Le Protocole PAAN

PAAN [Kaddoussi et al., 2009, Zgaya et al., 2008] est un protocole de négociation pour la gestion de crise développé au sein de notre laboratoire CRIStAL [2]. PAAN est basé sur des travaux antérieurs [Zgaya et Hammadi, 2007] où la question principale était la réaffectation des Workplan des agents mobiles (les routes) pour éviter les nœuds indisponibles dans un réseau de fournisseurs de services de transport. PAAN est très similaire à ANTS. Toutefois, son aspect original est le fait que les contrats peuvent être acceptés partiellement. Ceci veut dire qu'un contrat peut être divisé en plusieurs

2. http://www.cristal.univ-lille.fr/

sous-parties. Chaque sous-partie est étudiée séparément et l'accord peut être global (toutes les sous-parties sont acceptées) ou partiel (certaines sous-parties sont acceptées, d'autres sont refusées). PAAN présente deux couches de communication :

- *Couche HOTM*[3] *(un à plusieurs)* : pour des raisons de hiérarchie, l'initiateur (agent zone) responsable d'un ensemble de participants (agents zones de bas niveau) est le seul capable de lancer un processus de négociation de type HOTM.
- *Couche HMTM*[4] *(plusieurs à plusieurs)* : dans le cas d'une importante perturbation, l'aspect hiérarchique n'est plus respecté, et plusieurs agents zones de bas ou de haut niveau peuvent lancer des processus de négociation de type HMTM.

Dans ce travail, nous nous intéressons à la première couche du protocole HOTM. En effet, dans PAAN, le processus de négociation commence par un message *demande de modification* créé par un agent de gestion responsable des approvisionnements de la zone de crise (il est dans ce cas appelé initiateur de la négociation). Ce dernier demande aux agents gestionnaires des sous-zones de crise (appelés participants) s'ils peuvent modifier les dates de livraison choisies auparavant en raison d'un retard de livraison. En fonction de leur degré d'urgence, les participants répondent avec une *proposition de modification*. Ensuite, un contrat est proposé par l'initiateur que les participants peuvent refuser, accepter totalement ou accepter partiellement.

Les participants doivent également renvoyer un message de confirmation pour affirmer leur décision par rapport à ce contrat. Si le processus de négociation dépasse un délai bien défini, l'initiateur envoie un message *annulation* pour le terminer. La Fig. IV.6 illustre le diagramme de séquence du protocole PAAN.

### IV.5.2 Ontologies dans les SMA

Avec le développement des technologies d'information et de communication et les avancés des applications d'Internet, les informations sur Internet augmentent d'une manière exponentielle ; en même temps, les données sur Internet sont hétérogènes et dynamiques. Par conséquent, les utilisateurs d'Internet passent beaucoup de temps à chercher, collecter et maintenir les informations dont ils ont besoin. La recherche conventionnelle des informations sur la toile consiste à vérifier la correspondance entre les mots clés de la recherche et chaque mot dans les pages web et les documents. Cette méthode permet d'établir la correspondance (si elle existe) mais ne fournit aucune garantie sur la pertinence des réponses.

3. Help One-To-Many
4. Help Many-To-Many

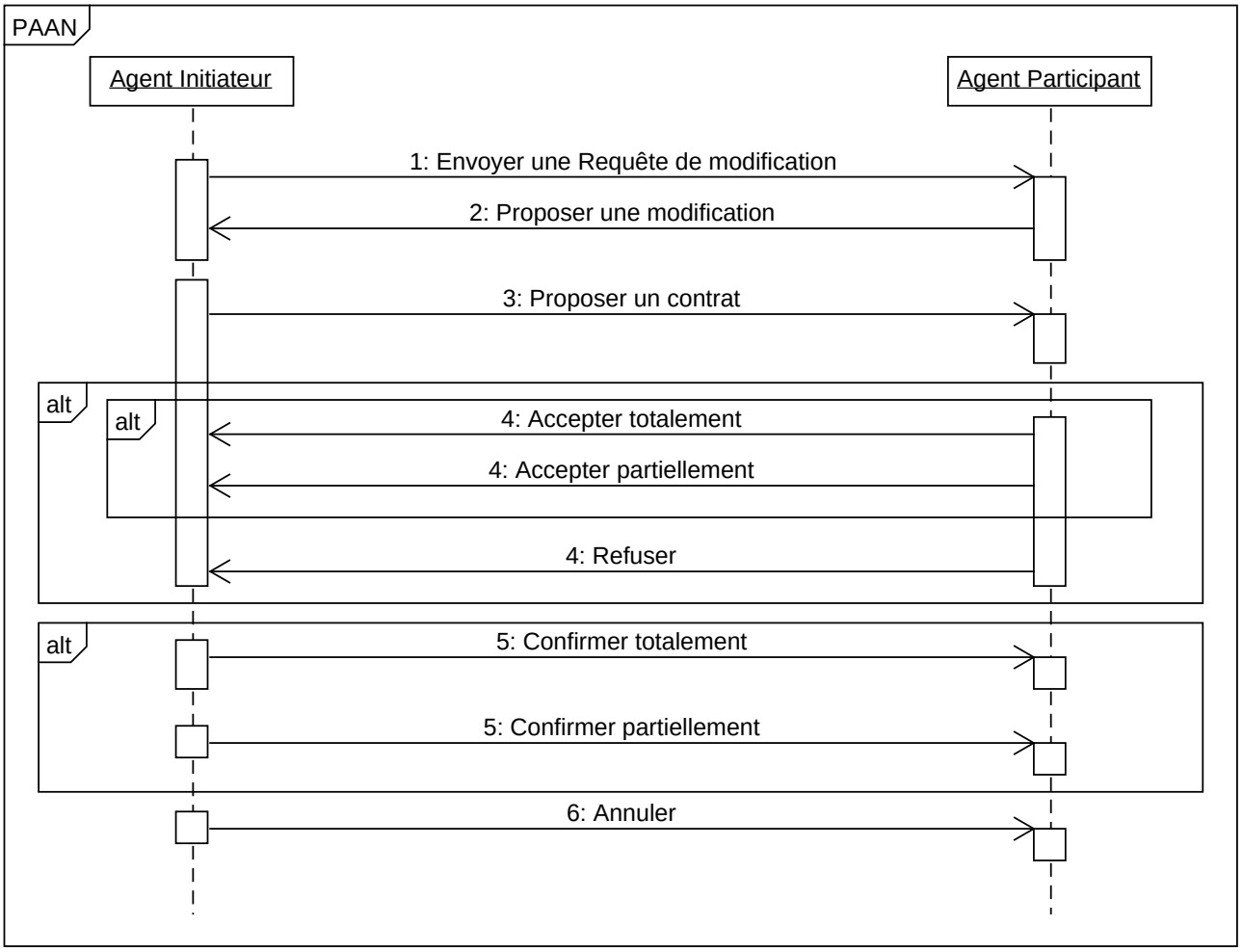



FIGURE IV.6: Illustration du diagramme de séquence du protocole PAAN.

L'introduction du web sémantique a pour objectif de créer des systèmes informatiques plus aptes à comprendre et satisfaire les demandes utilisateurs en modélisant les connaissances dans un domaine donné. La connaissance est alors structurée en un ensemble organisé de concepts, reliés par des liens sémantiques ainsi que par des liens de composition et d'héritage au sens objet. Une ontologie représente donc une organisation hiérarchique de la connaissance sur un ensemble d'objets par leur regroupement en sous-catégories suivant leurs caractéristiques essentielles.

Dans un SMA, lorsque deux agents communiquent à propos d'un objet, il est nécessaire pour eux d'être d'accord sur une terminologie qu'ils utilisent pour décrire cet objet. La communication s'effectue en s'envoyant des messages. Dans un message lié à un langage de communication donné, une information est représentée sous forme d'une expression de contenu, facile à transférer, codée avec le format approprié (ex. chaine de caractère, séquence d'octets, etc.) et conforme à un langage de contenu (ex. FIPA-ACL, KQML, OWL, KIF, etc.), qui peut être compréhensible ou pas par l'être humain. Cependant, chaque agent peut avoir sa propre disposition de représenter l'information qui

n'est pas toujours évidente à manipuler d'où la notion de l'ontologie qui a pour but de faciliter la manipulation des messages échangés entre agents. Par conséquent, une ontologie définit un vocabulaire et un ensemble de relations qui relient les différents éléments de ce vocabulaire, ce qui représente une solution adéquate au protocole de négociation proposé, qui dépasse les limites d'une communication agent traditionnelle.

Dans [Kang et Sim, 2011], un SMA de découverte de service dans le contexte d'un cloud est présenté. L'usage d'une ontologie permet aux utilisateurs du système de composer des requêtes sémantiques. L'ensemble de requêtes reçues par le système est répertorié sou forme d'une pile (ou une queue). Le traitement de ces requêtes s'effectue par le moyen d'un module de connexion qui permet de :

- sélectionner la requête à traiter
- évaluer la demande exprimée par cette requête. Cette tâche s'effectue en deux étapes. La première étape consiste à identifier les similarités sémantiques entre la demande et le dictionnaire sémantique du système. La deuxième étape consiste à établir la correspondance (appelée aussi "matching" ou alignement) entre les expressions ontologiques identifiées et les services proposés par le système ;
- filtrer les correspondances établies. En effet, plusieurs correspondances peuvent être établies par le système mais seulement certaines sont capables de répondre aux préférences de l'utilisateur ;
- recommander des services. En effet, certaines requêtes peuvent échouer au niveau de la tâche d'évaluation. Dans ce cas, le système essaye de rapprocher la demande utilisateur et effectuer des recommandations.

Un moteur de correspondance des ontologies appelé OntoMAS est proposé dans [Rajakaruna et al., 2012]. Ce moteur est basé sur la technologie agent et se compose de deux modules :

- un module agent composé de plusieurs types d'agents : Alignement Request Agent, Concept Resource Agent, String Matching Resource Agent, Structure Matching Resource Agent, Linguistic Matching Resource Agent et Upper Ontology Matching Resource Agent ;
- un module de connaissances qui permet de fournir le demaine de connaissance au premier module afin d'effectuer la tache de correspondance.

Le rôle d'OntoMAS consiste à récupérer en entrée deux ontologies issues de deux domaines différents pour générer en sortie le niveau de correspondance entre eux. Selon les expérimentations effectuées dans ce travail, OntoMAS est capable d'attendre un niveau de précision égal à 82.7%.

Un SMA de résolution des situations de malentendu est proposé dans [Saad et al., 2008]. L'objectif du système est d'améliorer le processus de négociation en proposant une ontologie généralisée qui permet à plusieurs agents utilisant des sémantiques hétérogènes de se comprendre. Le système se compose de trois couches :

– une couche de négociation : c'est le protocole de négociation qui définit la façon dont la négociation se produit entre les initiateurs et les participants ;
– une couche sémantique : elle consiste en un traducteur sémantique qui permet de générer les correspondances entre la sémantique d'un agent et l'ontologie du système ;
– une couche de gestion des connaissance : c'est la base de données qui répertorie les différentes classes des ontologies du système.

Dans la suite, nous présentons le protocole de négociation proposé. Nous détaillons ainsi ses spécifications et les stratégies de décision de l'initiateur et du participant lors du processus de négociation.

## IV.6 Protocole de Négociation Proposé

Pour offrir une couche de communication flexible et efficace aux membres du voisinage collaboratif, nous proposons un protocole de négociation multilatéral, multicritère et à accord partiel appelé MIPA ("Multi-lateral multi-Issue Partial-Agreement negotiation protocol"). Les deux caractéristiques essentielles de MIPA déduites de la table IV.10 sont :

– *Les ressources multiples* : c'est-à-dire que les mêmes services peuvent être récupérés depuis plusieurs fournisseurs de services (i.e. clients actifs) à différents coûts et délais de réponses (ex. $ids_4$ est proposé par $a_{16,1}$, $a_{12,1}$ et $a_{10,1}$) ;
– *L'accord partiel* : lorsqu'un demandeur de service trouve que l'ensemble des services demandés peut être récupéré depuis le même client actif, dans ce cas, un accord total ou partiel peut être établi pour récupérer la totalité de l'ensemble ou seulement une partie des services. Ceci dépend de la valeur de l'utilité de chaque proposition (ex. l'ensemble de services $\{ids_5, ids_{10}, ids_{11}\}$ peut être récupéré depuis le même client actif $a_{17,1}$).

### IV.6.1 Protocole de Négociation MIPA

Le protocole MIPA est une version améliorée de PAAN et adaptée à notre système. D'abord, MIPA est multilatéral puisqu'il implique au moins deux agents à chaque fois. Ces derniers négocient le coût global de chaque morceau d'information en termes de commodité pour chacun des deux agents négociateurs. Ensuite, il est automatisé puisqu'il ne demande pas l'intervention directe de l'utilisateur. Toutes les négociations sont exécutées à la place de l'utilisateur qui spécifie seulement les informations dont il a besoin ainsi que ses préférences. Les paramètres tels que le fournisseur, le coût ou la taille des

informations d'un service sont seulement déterminés par l'agent utilisateur qui cherche à satisfaire et à s'adapter aux préférences de l'utilisateur. La négociation est basée sur une stratégie d'accord partiel où l'initiateur a la possibilité d'acquérir un sous-ensemble du contrat et rejette le reste.

En reprenant la table de fournisseurs de services de l'exemple d'application présenté dans le paragraphe 1, nous pouvons constater qu'un même fournisseur de service (ex. $a_{5,1}$, $a_{12,1}$, $a_{16,1}$ ou $a_{17,1}$) peut fournir plusieurs services demandés par une même requête. En comparaison avec MIPA, des protocoles de la littérature tels que CNET [Smith, 1980] ou ANTS [Mathieu et Verrons, 2003] peuvent ne pas être très utiles pour chercher la meilleure offre puisqu'ils ne permettent d'établir que des accords totaux. C'est-à-dire qu'une requête demandant un ensemble de services ne peut être que refusée ou totalement acceptée par le fournisseur de services.

En effet, la négociation dans de tels protocoles, porte toujours sur un seul objet (*le contrat*) qui est souvent élémentaire insécable à chaque tour et qui peut être seulement totalement accepté ou refusé. Autrement dit, une annonce peut être faite que pour demander un ou plusieurs services mais sans qu'il y a la possibilité de récupérer une partie et non pas la totalité de l'ensemble des services.

Afin de surpasser ces limitations et garantir plus de flexibilité lors des échanges des informations entre les membres d'un voisinage collaboratif, nous proposons MIPA qui permet l'initiateur de négocier un ensemble d'objets de négociation (i.e. des services) simultanément et établir des accords partiels. Dans le paragraphe suivant, nous détaillons le modèle des différents messages de MIPA et nous présentons une description de chaque type de message.

#### IV.6.1.1 Modèle de Message dans MIPA

Le modèle d'un message MIPA est composé de 6 champs comme le montre la table IV.9. Le même modèle est utilisé par l'initiateur et le participant. Cependant, ce ne sont pas les mêmes champs que les négociateurs utilisent à chaque fois. A titre d'exemple, quand l'initiateur accepte une proposition, tous les champs sont utilisés à l'exception du champ contenu qui reste vide. Le champ date d'expiration est aussi utilisé seulement par l'initiateur dans le premier message d'annonce.

La Fig. IV.7 illustre un exemple d'un message MIPA envoyer par l'agent participant "userAgent-52-C1" à l'agent initiateur "userAgent-101-C1" pour proposer 4 services $\{ids_{15,1}, ids_{4,1}, ids_{23,1}, ids_{2,1}\}$.

TABLE IV.9: Les champs d'un message MIPA

| **Champs** | **Description** |
|---|---|
| Identifiant de communication | chaque message créé au cours d'un processus de négociation hérite du même identifiant de communication |
| Émetteur | l'identifiant de l'émetteur du message |
| Récepteur | l'identifiant du récepteur du message |
| Performative | l'action demandée par le message |
| Contenu | le contenu du message |
| Délai d'expiration | la date limite de réception des propositions |

```
messageID:      com4022
senderID:       userAgent-52-C1
receiverID:     userAgent-101-C1
performative:   PROPOSE
content:        {ids15-1, c1, d1}
                {ids4-1, c2, d2}
                {ids23-1, c3, d3}
                {ids2-1, c4, d4}
timeout:
```

FIGURE IV.7: Exemple d'un message MIPA.

#### IV.6.1.2 Description des Types de Messages dans MIPA

La description des messages dans MIPA est présentée dans cette partie. L'ontologie adoptée s'adapte avec le processus d'échanges des données entre les membres d'un voisinage collaboratif. Le diagramme de séquence de MIPA est présenté dans la Fig IV.8.

– *Message d'annonce ("Call for Proposal")* : un agent utilisateur demandant un service (ou un ensemble de services) donné, commence le processus de négociation en envoyant un message de type annonce aux fournisseurs de services (i.e. les clients actifs) identifiés. Le champ contenu du message est initialisé avec les identifiants des services demandés (i.e. objets de négociation). Pour chaque annonce, l'agent initiateur définit une date d'expiration (i.e. date limite d'envoi des propositions) que tous les participants doivent respecter afin d'effectuer des propositions valides ;

– *Message de proposition ("Propose")* : pour chaque objet de négociation représenté dans le contenu du message d'annonce, chaque participant génère une proposition qui consiste en un ensemble de valeurs qui correspondent à chaque critère (ex. coût, date de réponse, date de fin de validité, etc.). En effet, les agents utilisateurs sont capables de définir des coûts virtuels pour

les IMA qu'ils hébergent (§IV.6.3). Ceci leurs permet de profiter de plus de services. De plus, les utilisateurs seront motivés à héberger plus de services pour des durées de temps plus longues. L'hébergement des IMA est limité par une contrainte de temps fixé par le connecteur en se basant sur la classe de chaque service (§III.5.3), c'est la date de fin de validité. Dans ce cas, les agents utilisateurs cherchent à maximiser cette durée afin d'offrir le service à des durées plus longues.

– *Message d'accord total ("Total Accept")* : quand la proposition d'un participant garantit la meilleure valeur d'utilité pour chaque objet de négociation, dans ce cas, un accord total peut être établi et un message de type *Accepter Totalement* est envoyé au participant correspondant.

– *Message d'accord partiel ("Partial Accept")* : lorsqu'un participant donné offre la meilleure valeur d'utilité globale pour une partie des informations demandées, mais en même temps, d'autres propositions sont meilleures pour le reste, dans ce cas, un accord partiel est établi en envoyant un message de type *Accepter Partiellement*. L'agent initiateur spécifie les offres sélectionnées dans le contenu du message.

– *Message de rejet ("Reject")* : si aucune des offres d'une proposition n'est sélectionnée, dans ce cas, un message de type *Rejeter* est envoyé au participant correspondant.

– *Message d'information ("Inform")* : si une proposition est sélectionnée (suite à un accord total ou partiel), le participant renvoie un message de type *Informer* avec les informations demandées et le processus de négociation se termine.

– *Message de résiliation ("Terminate")* : au cours d'un tour de négociation, les agents participants sont obligés de respecter une certaine date spécifiée dans le message d'annonce au départ, afin d'effectuer des offres valides. A l'issue de cette date limite, l'initiateur envoie aux participants qui n'ont pas encore effectués des offres un message de type *Terminer* pour les informer qu'ils ne sont plus concernés par ce processus de négociation.

### IV.6.2 Algorithme de Décision de l'Initiateur

La table de fournisseurs de services générée à partir de l'algorithme d'identification des fournisseurs de services dans un voisinage collaboratif permet de créer une autre table : la table de négociation $NT^t_{h,j}$. Cette table permet de définir les objets de négociation qu'un initiateur de négociation utilise

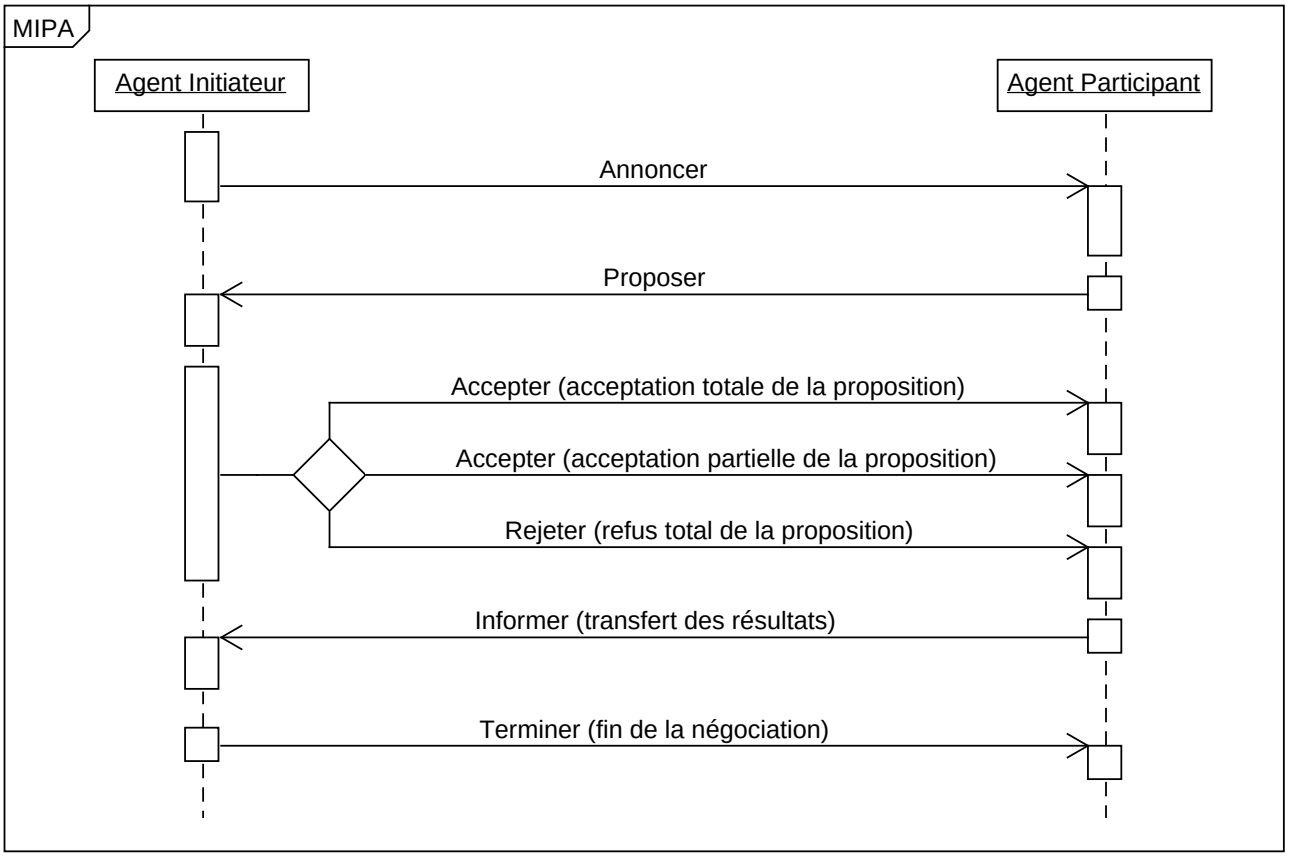



FIGURE IV.8: Illustration du diagramme de séquence du protocole MIPA.

pour effectuer des annonces. La table de fournisseurs de service de l'exemple du paragraphe IV.4.2 (table IV.8)peut être transformée pour donner la table IV.10.

TABLE IV.10: Exemple d'une table de négociation $NT_{7,1}^{100}$ à l'instant $t = 100$

| **Fournisseurs de services disponibles** | **Objets de négociation** |
|---|---|
| $a_{5,1}$ | $ids_5$, $ids_6$ |
| $a_{6,1}$ | $ids_6$ |
| $a_{8,1}$ | $ids_{10}$ |
| $a_{10,1}$ | $ids_4$ |
| $a_{12,1}$ | $ids_4$, $ids_6$ |
| $a_{13,1}$ | $ids_{10}$ |
| $a_{16,1}$ | $ids_4$, $ids_5$ |
| $a_{17,1}$ | $ids_5$, $ids_{10}$, $ids_{11}$ |
| $a_{25,1}$ | $ids_{11}$ |
| $a_{29,1}$ | $ids_{11}$ |
| $a_{32,1}$ | $ids_{13}$ |
| $a_{33,1}$ | $ids_{13}$ |
| $a_{34,1}$ | $ids_{13}$ |

Après l'élaboration de la table de négociation, l'agent utilisateur exécute l'algorithme afin de générer les offres et sélectionner les meilleures propositions.

**Algorithme 9** Algorithme de décision de l'initiateur

**Entrée:** $CT^t_{h,j}$
**Sortie:** *rep*
//Étape 1 : Annonce
Transformer $CT^t_{h,j}$ to $NT^t_{h,j}$
Initialiser un variable date limite $d$
Envoyer un message *ANNONCE* à chaque fournisseur de service
Lancer le Timer *timer*
//Étape 2 : traitement des propositions
**si** $timer > d$ **alors**
  Envoyer un message TERMINER aux agents n'ayant pas répondu
  **pour** chaque proposition **faire**
    Calculer la valeur d'utilité
  **fin pour**
  **pour** chaque fournisseur de service **faire**
    **si** toutes les propositions sont meilleure offre **alors**
      Envoyer un message ACCEPTER-TOTALEMENT
    **sinon si** certaines propositions sont meilleure offre **alors**
      Envoyer un message ACCEPTER-PARTIELLEMENT
    **sinon si** aucune proposition n'est meilleure offre **alors**
      Envoyer un message REJETER
    **fin si**
  **fin pour**
**fin si**
//Étape 3 : Traitement des messages de type INFORMER
Construire la réponse finale *rep*
**retourner** *rep*

**Objet de négociation** ($o_n$) : l'objet de négociation est l'ensemble de services qu'un agent utilisateur cherche à récupérer chez un ensemble d'agents participants noté par $P = \{\hat{a}_1, \hat{a}_2, .., \hat{a}_m\}$. L'ensemble d'objets de négociation est noté par $O = \{o_1, o_2, .., o_n\}$.

Supposons que chaque objet est évalué en se basant sur un ensemble de critères $K = \{1, 2, .., k\}$ (ex. coût, date de réponse, date de fin de validité). Nous supposons aussi que l'initiateur utilise une fonction linéaire d'utilité pour exprimer ses préférences. Cette fonction peut être formalisée comme suit :

$$U(X) = w_1 N_1(x_1) + w_2 N_2(x_2) + ... + w_k N_k(x_k) \tag{IV.2}$$

où $X$ représente une offre à $k$-attributs, $x_i$ est la valeur du $i^{ème}$ attribut, $w_i$ est le poids associé à l'$i^{ème}$ attribut avec $\sum_{i=1}^{k} w_i = 1$.

$N_i : R_+ \rightarrow [0, 1]$ est une fonction linéaire définie comme suit :

$$N_i(x_i) = \frac{x_i^{max} - x_i}{x_i^{max} - x_i^{min}} \tag{IV.3}$$

où $x_i^{max}$ et $x_i^{min}$ sont respectivement les valeurs maximales et le minimales du $i^{ème}$ attribut.

### IV.6.3 Algorithme de Décision du Participant

Dans cette partie, nous détaillons la stratégie de décision d'un agent utilisateur participant dans un processus de négociation. Nous présentons aussi l'algorithme de décision du participant qui permet de mettre en œuvre cette stratégie.

#### IV.6.3.1 Stratégie de Décision

L'algorithme de soumission des propositions adoptée par les participants se base sur une stratégie de type *participant-impatient*. En effet, cette stratégie consiste à proposer des coûts de services qui diminuent en fonction du temps. Pour cela, nous définissons les paramètres suivants :

**Fenêtre de temps** $[t_0, DV_{m,j}]$ : rappelons que $DV_{m,j}$ représente la date de fin de validité d'un service $ids_m$. L'instant $t_0$ représente le moment auquel le participant (le client actif) proposant le service $ids_m$ l'a reçu.

**Facteur de remise** $(\xi_{h,j})$ : le facteur de remise noté par $\xi_{h,j}$ représente le coefficient d'amortissement des coûts des services qu'un client actif $\hat{a}_{h,j}$ propose. $\xi_{h,j} \in [0, 1]$. Le facteur de remise définit le niveau d'impatience (ou de patience) d'un agent participant.

**Coût initial** $(ci_m)$ **et coût proposé** $(cp_{m,h}^t)$ : le coût initial d'un service $ids_m$, noté par $ci_m$, est le coût du service lors de sa récupération depuis le connecteur ou d'un client actif . Le coût proposé par un client actif $\hat{a}_{h,j}$ pour un service $ids_m$, noté par $cp_{m,h}^t$, est un coût virtuel qu'un agent participant affecte à un service demandé par un agent initiateur lors d'un tour de négociation à l'instant $t$. Le coût proposé est calculé en se basant sur le coût initial et le facteur de remise.

**Remise** $(\xi_{h,j}^t)$ : la remise notée par $\xi_{h,j}^t$ représente la valeur de l'amortissement du coût initial d'un service $ids_m$ proposé par un client actif $\hat{a}_{h,j}$ à l'instant $t$. $\xi_{h,j}^t \in [0, 1]$.

Cette stratégie implique que les agents ne peuvent vendre les services qu'à des coûts inférieurs ou égaux aux coûts initiaux. En effet, en se basant sur la remise $\xi_{h,j}^t$, la proposition d'un coût d'un service $ids_{m,j}$ à un instant $t$ est générée par la formule suivante :

$$cp_{m,h}^t = \xi_{h,j}^t \times ci_m = (\xi_{h,j})^t \times ci_m \tag{IV.4}$$

La Fig. IV.9 illustre un exemple de l'évolution du coût proposé pour des différentes valeurs du facteur de remise : $\xi_{h,j} \in \{1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.2, 0.05\}$, avec $ci_m = 10$ et $DV_{m,j} = 10$.

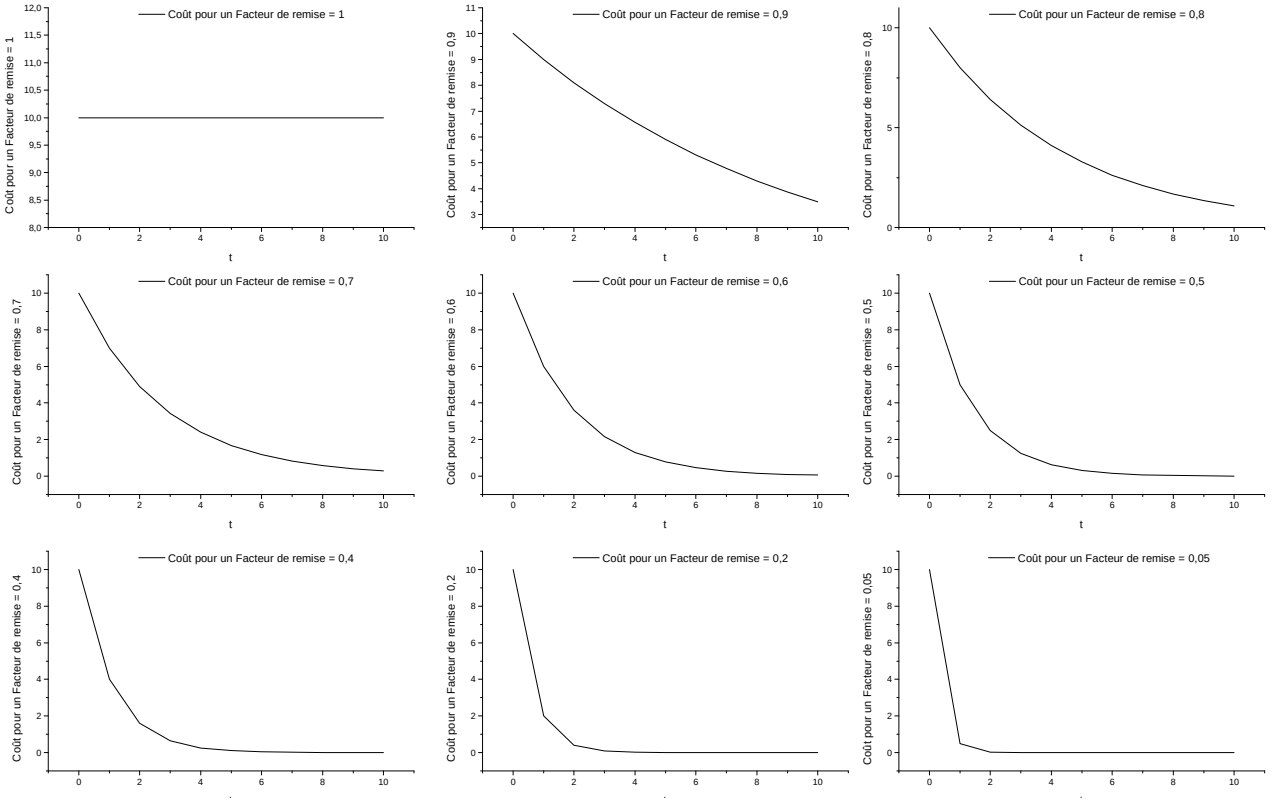



FIGURE IV.9: Exemple de l'évolution du coût proposé pour des différente valeurs du facteur de remise. Le coût initial est égal à 10 et la durée de validité est égale à 10.

D'après la Fig. IV.9, lorsque le facteur de remise diminue, la valeur du coût proposé diminue plus rapidement. D'où le sens d'impatience exprimé dans cette stratégie. En effet, un agent participant très impatient tente de diminuer le coût des services qu'il propose plus rapidement afin d'effectuer un maximum de ventes avant la fin de la durée de validité d'un service donné. La Fig. IV.10 illustre lu degré d'impatience d'un agent participant.

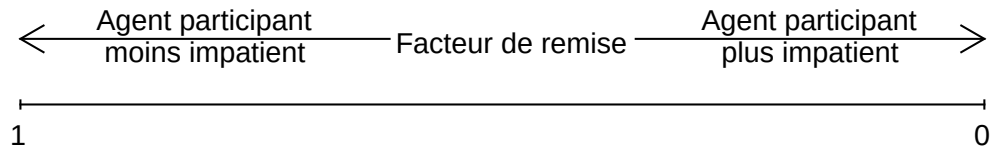



FIGURE IV.10: Illustration du degré d'impatience d'un agent participant.

#### IV.6.3.2 Algorithme de Décision du Participant

L'algorithme de décision du participant (algorithme 10) s'exécute en deux étapes :

– $1^{ère}$ étape : consiste à traiter les messages d'annonces et générer les propositions

– $2^{ème}$ étape : consiste à traiter les messages d'accords ou de refus et renvoyer les informations demandées.

**Algorithme 10** Algorithme de décision du participant

**Entrée:** Les coûts initiaux, le facteur de remise, les dates limites de validité
**Sortie:** *rep*

```
    //Étape 1 : Traitement des messages de type ANNONCE
 1: Initialiser un Timer timer
 2: Récupérer la date limite de réception des proposition timeout
 3: Récupérer l'annonce A
 4: pour chaque service dans A faire
 5:    si service est encore valide alors
 6:       Calculer le coût proposé
 7:       Créer une proposition avec le coût proposé et la date limite de validité
 8:       Ajouter la proposition à la liste des proposition
 9:    fin si
10: fin pour
11: si timer ≤ timeout alors
12:    Envoyer la liste des propositions à l'agent initiateur
13: fin si
    //Étape 2 : Traitement des messages de types ACCORD-TOTAL/PARTIEL et REFUSER
14: si ACCORD-TOTAL alors
15:    Générer l'ensemble des informations demandées rep
16: sinon si ACCORD-PARTIEL alors
17:    Générer l'ensemble des informations demandées rep
18: sinon
19:    Ne rien faire
20: fin si
21: retourner rep
```

## IV.7 Conclusion

Dans ce chapitre, nous avons présenté l'approche orientée rôle pour la gestion des services dans un voisinage collaboratif. Les agents utilisateurs peuvent demander les services depuis le connecteur qui optimise la recherche et délivre les réponses. Une fois qu'un agent utilisateur récupère la réponse à sa demande, il peut avoir aussi la permission de changer son rôle et devenir un client actif afin de fournir les données qu'il vient de recevoir. L'affectation des permissions est basée sur des indices de rareté et de popularité qui ont pour rôle de garder un niveau suffisant de redondance des services au

sein du voisinage collaboratif. Pour échanger les services, les agents utilisateurs communiquent selon le protocole MIPA qui leur permet d'améliorer la flexibilité des accords établis.

A ce stade, nous avons présenté les approches développées dans ce travail de thèse et nous avons construit une plateforme de recherche et de composition des services d'aide à la mobilité adaptée pour les espaces ubiquitaires. La validation de ces approches est démontrée avec les résultats de simulation dans le chapitre suivant.

# Chapitre V

# Simulation et Analyse des Résultats

**Sommaire**



## V.1 Introduction

Dans ce dernier chapitre, nous présentons la simulation et les résultats des approches développées dans cette thèse. Nous présentons en premier lieu les spécificités de la programmation orientée agent,

en particulier, nous présentons la différence entre un le paradigme objet et le paradigme agent. Nous soulignons aussi la technologie agent mobile et nous présentons quelques outils et plateformes de développement des applications orientées agent. Nous concluons ainsi quant aux choix de la plateforme de développement de l'application de simulation. Nous présentons en deuxième lieu les caractéristiques de la plateforme de développement adoptée. Ainsi nous évoquons les différents outils de paramétrage et les spécifications de développement. Ensuite nous présentons l'outil de simulation développé en se basant sur les approches de modélisation et d'optimisation adoptées. Nous présentons ainsi la vue d'ensemble, les outils de configuration et quelques limitations et axes d'améliorations possibles. Finalement, nous présentons des scénarios de simulation réalisés à l'aide de cet outil.

## V.2 Programmation Orientée Agent

Partant des caractéristiques spécifiques aux agents et aux SMA qui démontrent la nécessité de créer des nouveaux outils capables de supporter ce nouveau paradigme, la programmation orientée agent (POA) représente selon certains un atout qui va renouveler la manière avec laquelle les logiciels et les applications sont conçus et développés [Lind, 2001, André Filipe de Moraes et al., 2011, Boissier, 2013]. Dans ce paragraphe, Nous soulignons aussi la différence entre un agent et un objet. Nous présentons un type très spécifique d'agents : les agents mobiles, nous présentons enfin quelques outils de développement des SMA et nous argumentons notre choix de la plateforme de développement de notre application.

### V.2.1 Agent vs Objet

En plus d'être un outil performant d'abstraction et de modélisation des systèmes distribués, le paradigme agent possède aussi un aspect applicatif et révèle souvent la question concernant son apport par rapport aux outils de programmation existant, en particulier la POO. Quelques éléments de réponse à cette question ainsi que les outils de développement des SMA sont présentés dans ce qui suit.

Qu'est un objet ? En programmation structurée, un programme est formé de différentes procédures et de structures de données indépendantes de ces procédures. C'est à dire que l'accès à ces données peut être effectué directement. En programmation orientée objet, un programme met en œuvre différents objets. Chaque objet associe des données (états) à des méthodes (actions) agissant exclusivement sur les données de l'objet. Notons qu'on utilise le mot méthode au lieu de procédure et ceci n'est pas une simple substitution. En effet, le concept de base en orienté objet est ce qu'on appelle l'encapsulation des données. Cela signifie que les données d'un objet sont généralement accessible à l'aide des méthodes

de cet objet ; il est nécessaire de passer par ses méthodes qui jouent ainsi le rôle d'une interface obligatoire.

Dans la vie réelle, un objet est l'équivalent de n'importe quel objet physique qui ne peut être manipulé qu'à travers ses interfaces (ex : un interrupteur ; il faut que quelqu'un appuie dessus pour allumer la lumière). Nous constatons que l'objet est une entité passive, qui ne peut pas décider de son fonctionnement, une fois qu'une de ses méthodes est appelée, il n'a qu'à répondre à cet appel.

Considérons un système multi-agent, où chaque agent possède son propre agenda et ses propres objectifs ; nous ne pouvons pas garantir l'exécution d'une action $m_1$ par un agent $a_1$ juste parce qu'un autre agent $a_2$ lui a demandé de le faire ; $m_1$ peut ne pas être dans l'intérêt de $a_1$. Ce degré d'autonomie vis-à-vis du choix d'exécuter ou non une action, représente la différence fondamentale entre un objet et un agent. Dans le cas d'un objet, la décision demeure à l'objet qui appelle la méthode. Dans le cas d'un agent, la décision demeure à l'agent qui reçoit l'appel. D'après Wooldridge [Woolridge, 2001], un slogan a été proposé pour résumer et répondre à la question : *Objects do it for free ; agents do it because they want to* (Les objets le font gratuitement ; les agents le font parce qu'ils le veulent). Le tableau V.1 récapitule la comparaison entre un agent et un objet.

TABLE V.1: Comparaison entre les caractéristiques d'un agent et d'un objet

| **Entité** | **Objet** | **Agent** |
|---|---|---|
| Encapsulation des états internes (ex. constantes et variables privées) | oui | oui |
| Autonomie (exécution de l'action spécifiée dans un message) | non (un objet répond toujours à l'appel de ses méthodes) | oui (un agent décide librement de répondre ou non à l'appel reçu) |
| Comportement | passif (il ne réagit que lorsqu'il reçoit un message) | proactif (il cherche à atteindre ses buts en saisissant les opportunités qui s'offrent à lui) |
| Sociabilité | non | oui (capable de s'engager dans des interactions plus ou moins complexes : coopération, négociation, collaboration) |
| Type de message | pas de spécifications | modélisés à partir de la théorie des actes (accepter, refuser, informer, confirmer, etc.). Langages connues : FIPA ACL13, KQML, KIF |

### V.2.2 Outils de Programmation Orientée Agent

La plateforme multi-agents est le support qui permet la création, la manipulation et l'expérimentation d'un système multi-agents. Elle se caractérise par des modèles de mise en œuvre, des procédures d'interaction et de communication et aussi par la modélisation des environnements et des connaissances.

Plusieurs plateforme de développement des SMA sont disponibles telles que : JADE [1], MaDKit [2], ZEUS [3], JANUS [4], AgentBuilder [5], StarLogo [6], NetLogo [7], IBM AGLETS [8], etc. Dans [Nikolai et Madey, 2009], Nikolai présente une classification détaillée des outils de développement des SMA.

Le choix de l'outil adéquat doit répondre au maximum des critères suivants : disponibilité, configuration, documentation, applications, hétérogénéité, adaptabilité, standards, interopérabilité, distribution, portabilité, facilité de développement, facilité d'expérimentation, mécanismes de simulation, mobilité des agents, outils graphiques, open source. La classification des outils de développement selon les critères précédents et selon les domaines d'application de chaque plateforme fait partie de nos perspectives.

### V.2.3 Choix de l'Outil de Développement

Certains outils imposent des méthodologies spécifiques pour le développement des applications orientées agents. D'autres sont primitifs, fermés et ne dispose pas d'une flexibilité de développement suffisante. Les outils de développement qui nous intéressent sont : JADE, JANUS et ZEUS.

ZEUS permet un développement des applications orientées agent en langage Java. ZEUS offre une librairie de composants et en propose un environnement de développement visuel. En effet, ZEUS propose des interfaces graphiques de saisie des spécifications utilisateur et de génération automatique des codes en se basant sur une collection de classes qui forment la structure de base des agents dans ZEUS. Un agent dans ZEUS est composé de cinq couches : couche API, couche définition, couche organisationnelle, couche de coordination et couche de communication.

1. http://jade.tilab.com/
2. http://www.madkit.net/madkit/
3. http ://sourceforge.net/projects/zeusagent/http ://sourceforge.net/projects/zeusagent/
4. http://www.janus-project.org/Home
5. http://www.agentbuilder.com/
6. http://education.mit.edu/starlogo/
7. http://ccl.northwestern.edu/netlogo/
8. http://aglets.sourceforge.net/

JANUS est une plateforme open-source de développement des applications orientées agent implémentée en langage Java. Cette plateforme est optimisée pour les systèmes multi-agents de type organisation holonique et se base sur la méthodologie ASPECS [Cossentino et al., 2009]. JANUS propose des outils pour l'exécution, l'affichage et la gestion des applications à base d'agents. Les applications développées avec JANUS peuvent être distribuées sur le réseau.

JADE (Java Agent Development Environment) est une plateforme de développement des applications orientées agent en langage Java. JADE permet une interopérabilité flexible des systèmes multi-agents communicants grâce à l'implémentation du langage ACL [9], conforme aux spécifications de la FIPA [10], pour le transfert des messages. JADE est un projet open-source et offre des composants de développement des agents légers (capable de s'exécuter sur des terminaux mobiles) grâce à la librairie LEAP. Depuis la version 4.0, JADE et JADE-LEAP deviennent une seule plateforme. JADE dispose de plusieurs caractéristiques :

- offre une plateforme conforme aux spécifications FIPA qui inclut trois agents : AMS, DF et ACC. Ces agents sont démarrés automatiquement au lancement de la plateforme ;
- dispose d'une plateforme distribuée qui peut être divisée sur plusieurs machines. Les agents sont implémentés en tant que processus Java et la communication entre des agents d'une même machine est assurée grâce aux événements Java ;
- offre une communication inter-agent effectuée par le moyen des messages ACL qui sont représentés par des objets Java pour des échanges plus flexibles et légers ;
- supporte la mobilité des agents à travers le réseau au sein d'une même instance de plateforme ;
- offre une librairie de gestion des ontologies définies par le développeur ;
- offre une interface et un agent (l'agent RMA) pour la gestion de la plateforme et des agents. Cette interface offre la possibilité de créer des nouveaux agents, les suspendre, les transférer, les terminer, etc. ;
- supporte l'intégration des web services, l'invocation dynamique des web services, l'intégration des Applet ;
- dispose d'un ensemble d'agents de monitoring de la plateforme tels que le DummyAgent, le Sniffer, le DF GUI et l'IntrospectorAgent ;
- offre la possibilité d'implémenter des systèmes relativement complexes ;
- ne spécifie pas une méthodologie de développement particulière des systèmes multi-agents ;
- permet l'accélération de développement grâce à la présence suffisamment importante de briques logicielles pour pouvoir produire une application aboutie ;

9. Agent Communication Language
10. Foundation for Intelligent Physical Agents [http://www.fipa.org/]

– supporte la portabilité grâce à l'utilisation de Java comme langage de programmation.

Par conséquent, pour le développement de notre système et la simulation des résultats de nos approches d'optimisation distribuée, nous avons choisi la plateforme JADE. Dans la suite, nous présentons les différentes caractéristiques de la plateforme JADE, en particulier les langage de communication FIPA-ACL.

## V.3 Caractéristiques de la Plateforme de Développement

### V.3.1 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 (Fig. V.1). Pour chaque plateforme il existe toujours un et un seul conteneur principal appelé *main container*. Ce conteneur héberge l'AMS (Agent Management System), le DF (Directory Facilitator) et l'ACC (Agent Communication Channel). Tous les autres conteneurs (appelés auxiliaires) doivent se déclarer dans le main container. Les conteneurs auxiliaires peuvent être démarrés dans des machines distantes et gérés à distance à travers le réseau.

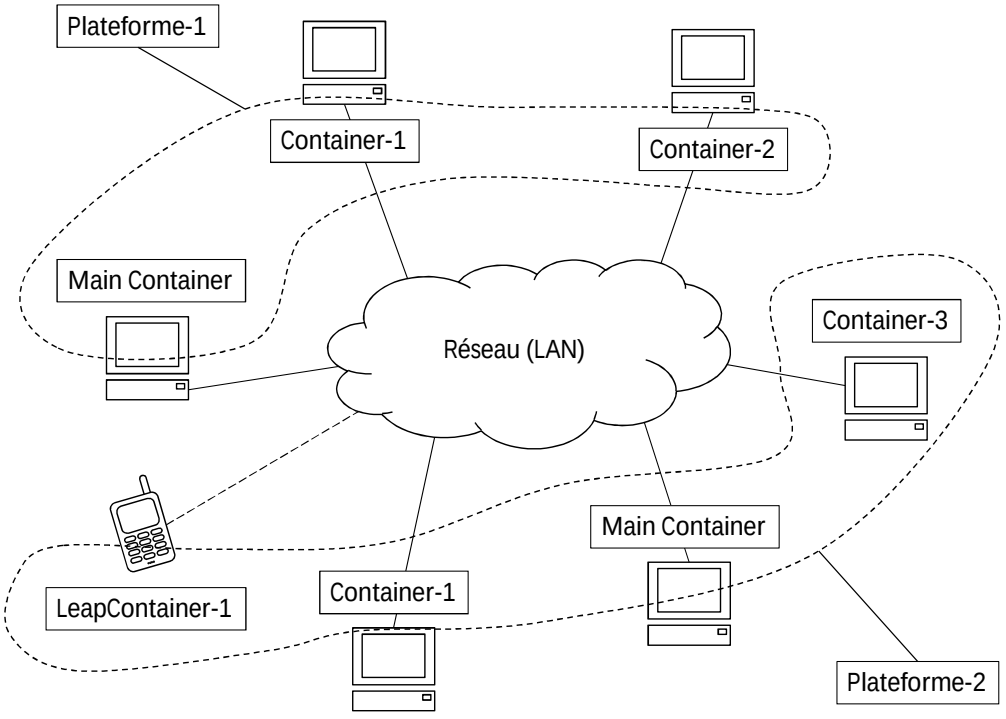



FIGURE V.1: Illustration de la plateforme JADE.

### V.3.2 Outils de Gestion de la Plateforme

JADE offre des outils utiles à la gestion de la plateforme et au débogage :

*Dummy Agent*

– visualisation des messages ;

– envoi des messages aux agents présents sur la plateforme et réception des réponses.

*Sniffer Agent*

– visualisation de l'enchaînement des messages, et des messages eux mêmes ;

– vérification interactive du déroulement des communications.

*Introspector Agent*

– visualisation des messages envoyés et reçus ;

– visualisation des comportements actifs et non actifs ;

– contrôle des état de l'agent.

### V.3.3 Spécifications du Langage de Communication

Le langage de communication ACL est un langage haut niveau issu de la théorie des actes. Il prend 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 [11] 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 encodés dans un agent de langage de communication ACL. Nous avons listé quelques-unes des communications possibles les plus utilisées et les avons regroupées dans la table V.2.

Le message type du Fipa ACL contient tout d'abord le type du message envoyé (c'est-à-dire syntaxe de ce message), mais aussi :

– l'expéditeur du message ;

– le destinataire du message ;

– l'identifiant de la communication ;

– le contenu du message ;

– le langage utilisé dans le contenu du message ;

– le protocole utilisé ;

– l'ontologie à laquelle le message se rattache ;

11. Knowledge Query and Manipulation Language

Table V.2: Principales actions de communication dans FIPA-ACL

| **Actions** | **Syntaxe** | **Description** |
|---|---|---|
| Accepter une offre | ACCEPT-PROPOSAL | Communication de l'accord de l'expéditeur d'effectuer une action qui lui a été préalablement soumise |
| Approuver | AGREE | Communication de l'accord de l'expéditeur pour effectuer une action, sans doute dans le futur |
| Annuler | CANCEL | Communication de l'annulation de l'accord donnée préalablement par l'expéditeur pour effectuer une action |
| Appel d'offre | CFP | Communication par l'expéditeur d'une demande d'effectuer une certaine action |
| Confirmer | CONFIRM | Communication par l'expéditeur de la confirmation de la validité (selon les règles de l'agent) de la proposition préalablement reçue |
| Dénier | DISCONFIRM | Communication par l'expéditeur de la confirmation de la non validité (selon les règles de l'agent) de la proposition préalablement reçue |
| Informer | INFORM | Communication par l'expéditeur d'une proposition, pensée vrai par celui-ci |
| Proposer | PROPOSE | Communication par l'expéditeur d'une proposition d'action conditionnée à certaines préconditions données |
| Demander | REQUEST | Communication par l'expéditeur d'une demande au destinataire d'effectuer une action |

– la référence d'un message antérieur auquel le message actuel se rattache, ou la référence d'un message ultérieur attendu en retour ;

## V.4 Web Services et Transfert des Résultats

Un web service, est un programme informatique qui tourne sur un serveur, qui calcule les réponses à des requêtes reçues par le serveur, et renvoie des réponses à ces requêtes assurant ainsi l'échange de

données entre systèmes différents. A titre d'exemple, un service web "convertisseur de devises", qui aurait une fonction "EUR-TND" [12] prenant en entrée une devise en euro, la convertirait et rendrait en dinars tunisien. C'est la partie *serveur* du service web. La partie *client* exécutée sur un autre ordinateur connecté au réseau, enverrait une requête à la partie serveur : il suffit pour cela de connaître l'adresse du serveur, le nom de la fonction et son utilisation, ainsi que l'adresse d'un fichier WSDL [13]. Une fois une requête est parvenue au serveur, ce dernier génère un ensemble de données qui peuvent être contenues dans un fichier de données de type XML [14] et les renvoie au client. La figure V.2 [15] présente un exemple d'un fichier XML d'un web service de météo.

Ce fichier WSDL est un fichier, respectant les normes XML, qui décrivent toutes les modalités de l'échange entre le client et le serveur : description des fonctions disponibles, du type de données qui transitent, etc. Ce fichier est situé sur le serveur qui contient la partie *serveur* du service web. Lorsqu'un client envoie une requête vers la partie serveur, on dit qu'on *consomme* le service web. Ainsi, on peut voir le web service (partie client) comme une boîte noire : on lui envoie des informations en entrée, et on reçoit d'autres informations en sortie. Le tout est totalement opaque : il n'y a aucun moyen de savoir ce qu'il y a dans la boîte, quel langage est utilisé, quelle technologie est mise en œuvre, etc.

La partie *client* du web service est en général assez courte : quelques lignes de code servent à appeler le web service, et le reste consiste principalement à trier les données reçues dans la réponse, et les afficher correctement à l'utilisateur. L'intérêt du web service est que ce système facilite grandement les échanges entre diverses applications, sans se soucier d'une possible incompatibilité entre plusieurs langages.

En effet, en théorie, il n'existe presque aucune contrainte quant aux langages utilisés, que ce soit dans la partie client ou serveur. Principalement, il faut que ces langages rendent possible l'utilisation de web services. Nous pouvons ainsi, par exemple, avoir un web service (partie serveur) codé en Java, qui communique avec plusieurs clients, un en .NET, l'autre en Java, l'autre en PHP, etc.

Les web services dans ce cas peuvent être caractérisés par des temps de réponse, des quantités de données et des coûts. Dans le système proposé dans cette thèse, les terminaux mobiles des utilisateurs sont capables de stocker des fichiers de données (ex. les fichier XML) pour pouvoir les échanger ensuite les un avec les autres en se basant sur les tables d'état des voisinages collaboratifs auxquelles ils appartiennent.

12. Euro vers Dinar Tunisien
13. Web Services Description Language (norme des descripteurs d'un service web)
14. Extensible Markup Language : c'est un langage à base de balises qui permet de hiérarchiser des informations
15. http://w1.weather.gov/xml/current_obs/

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="latest_ob.xsl" type="text/xsl"?>
<current_observation version="1.0"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation="http://www.weather.gov/view/current_observation.xsd">
    <credit>NOAA's National Weather Service</credit>
    <credit_URL>http://weather.gov/</credit_URL>
    <image>
        <url>http://weather.gov/images/xml_logo.gif</url>
        <title>NOAA's National Weather Service</title>
        <link>http://weather.gov</link>
    </image>
    <suggested_pickup>15 minutes after the hour</suggested_pickup>
    <suggested_pickup_period>60</suggested_pickup_period>
    <location>Unknown Station</location>
    <station_id>41009</station_id>
    <observation_time>Last Updated on Apr 14 2015, 4:20 am EDT</observation_time>
        <observation_time_rfc822>Tue, 14 Apr 2015 04:20:00 -0400</observation_time_rfc822>
    <temperature_string>76.1 F (24.5 C)</temperature_string>
    <temp_f>76.1</temp_f>
    <temp_c>24.5</temp_c>
    <water_temp_f>77.0</water_temp_f>
    <water_temp_c>25.0</water_temp_c>
    <wind_string>South at 4.5 MPH (3.89 KT)</wind_string>
    <wind_dir>South</wind_dir>
    <wind_degrees>190</wind_degrees>
    <wind_mph>4.5</wind_mph>
    <wind_gust_mph>0.0</wind_gust_mph>
    <wind_kt>3.89</wind_kt>
    <pressure_string>1020.0 mb</pressure_string>
    <pressure_mb>1020.0</pressure_mb>
    <dewpoint_string>69.4 F (20.8 C)</dewpoint_string>
    <dewpoint_f>69.4</dewpoint_f>
    <dewpoint_c>20.8</dewpoint_c>
    <mean_wave_dir>East</mean_wave_dir>
    <mean_wave_degrees></mean_wave_degrees>
    <disclaimer_url>http://weather.gov/disclaimer.html</disclaimer_url>
    <copyright_url>http://weather.gov/disclaimer.html</copyright_url>
    <privacy_policy_url>http://weather.gov/notice.html</privacy_policy_url>
</current_observation>
```

FIGURE V.2: Exemple d'un fichier XML d'un web service de météo.

Dans le suite, nous présentons l'outil que nous avons développé et appelé MASiM [16].

## V.5 Présentation de l'Outil MASiM

MASiM est un outil de simulation des performances d'un système distribué à base d'agents intelligents dans un espace ubiquitaire. Les utilisateurs du transport ont besoin d'informations de mobilité avancée

16. 3L Multi-Agent System SiMulator. (3L correspond au trois lettres "L" dans le nom de la ville de Lille)

qui doivent être optimisées et pertinentes. L'application MASiM a pour but de simuler les résultats des approches proposées dans cette thèse. Dans cette partie du chapitre, nous présentons un vue d'ensemble de MASiM. Nous présentons ensuite les différents outils de configuration et de gestion des scénarios de simulation. Enfin, nous discutons les limitations de cet outil et nous proposons des axes d'amélioration.

### V.5.1 Vue d'Ensemble

Au lancement de l'application, la fenêtre de supervision (V.3) s'affiche avec un guide d'utilisation rapide (n'est pas affiché dans le figure). Cette fenêtre permet d'accéder aux différents outils de configuration des scénarios de simulation. Cette interface permet d'afficher d'une façon dynamique les messages et les résultats de communication des agents du système. Elle permet aussi grâce à un menu contextuel de :

- réinitialiser la console de sortie ;
- copier les données de communication de la console de sortie ;
- afficher et masquer l'outil de gestion de la plateforme JADE : les agents RMA, SNIFFER et INTROSPECTOR ;
- imprimer rapidement les données de communication de la console de sortie ;
- afficher le menu d'aide ;
- afficher différentes informations sur l'outil, la plateforme de développement et la machine hôte.

De plus, elle permet d'afficher l'évolution des scénarios en cours d'exécution grâce à une barre de progression. Plusieurs types de sorties sont imprimés par les agents du système sur la console de sortie :

- état d'un agent (démarré ou arrêté) ;
- tâche en cours d'exécution ;
- envoie d'un message ;
- réception d'une demande ;
- composition d'une requête ;
- recherche des fournisseurs de services ;
- temps de réponse à une requête ;
- mise à jour d'une table de service ;
- etc.

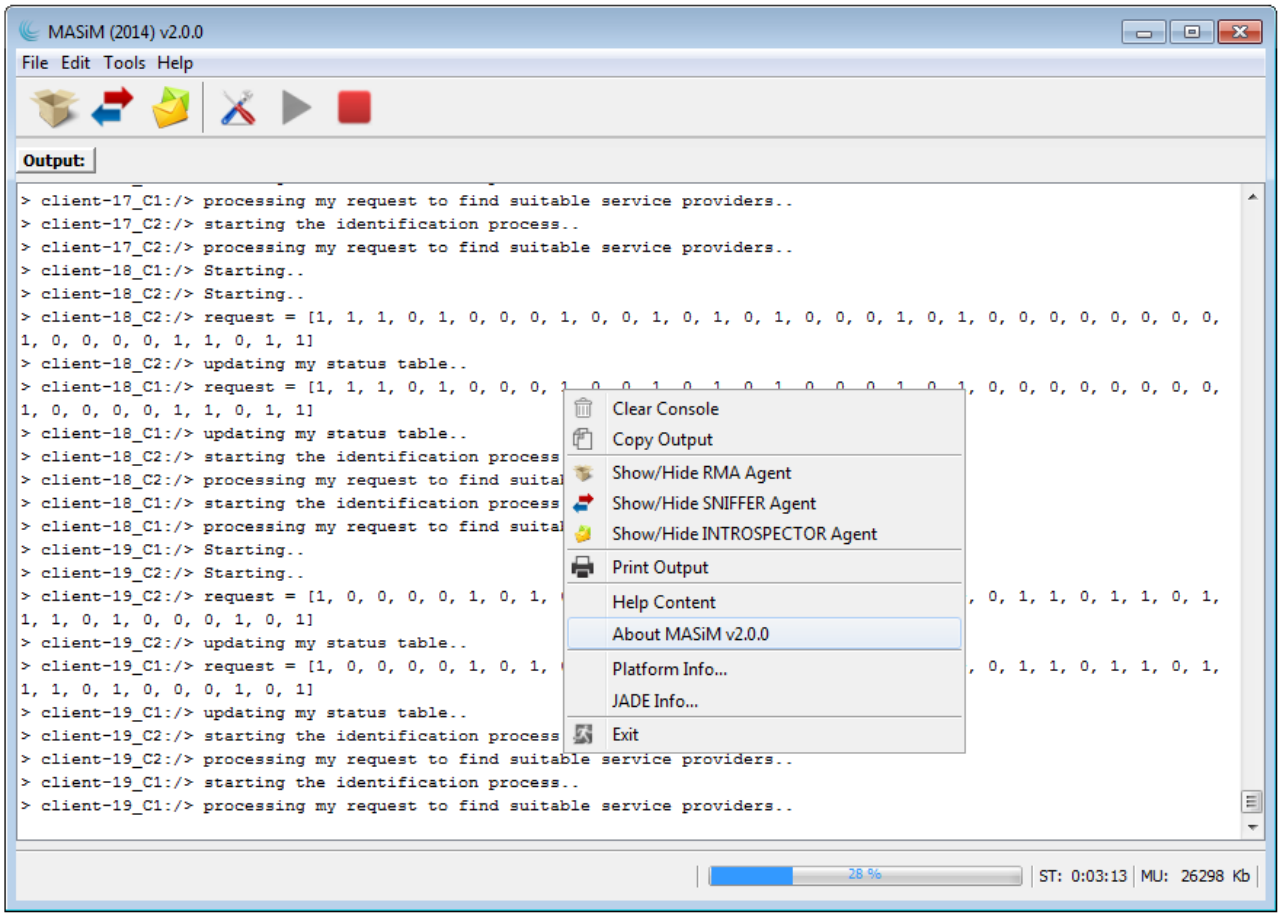



FIGURE V.3: Illustration de l'interface principale de supervision de l'outil MASiM.

Un extrait de la console de MASiM est illustré dans l'annexe D de ce mémoire. En outre, la fenêtre de supervision offre un menu de contrôle très simple composé de :

- un bouton de configuration des scénarios de simulation ;
- un bouton de démarrage des simulations ;
- un bouton d'arrêt des simulations ;
- un bouton pour lancer/masquer l'agent RMA ;
- un bouton pour lancer/masquer l'agent SNIFFER ;
- un bouton pour lancer/masquer l'agent INTROSPECTOR.

Après la configuration d'un scénario (ou plusieurs), MASiM dispose d'autres fenêtres de supervision (chaque pair connecteur, voisinage collaboratif possède une fenêtre indépendante) illustrées dans la Fig. V.4 et V.5. En effet, cette fenêtre dispose de quatre onglets :

- *Onglet 1 : Vue dynamique du système ("Dynamic System Overview").* Cet onglet permet d'afficher quelques informations statistiques telles que le nombre d'agents utilisateurs, le nombre de clients

actifs, le nombre de services demandés, etc. d'une manière dynamique. Il permet aussi d'afficher l'évolution dynamique de la position géographique des agents utilisateurs, le rôle des agents (un rôle actif est associé à un visage avec un chapeau avec une couleur d'écriture verte ; un rôle passif est associé à un visage sans chapeau et une couleur d'écriture bleue), les différentes informations relatives à chaque agent (identifiant de l'agent, nombre de service demandé, nombre de service fournis). La Fig. V.4 présente une illustration de l'onglet de la vue dynamique du système ;

– *Onglet 2 : Table d'état ("Status Table Overview")*. Cet onglet permet d'afficher l'évolution en temps réel de la table d'état dans le voisinage collaboratif. Nous rappelons que les tables d'état sont des tables distribuées dans le voisinage collaboratif afin de permettre aux agents utilisateurs d'identifier les fournisseurs de services (i.e. les clients actifs). Cet onglet permet d'afficher l'historique des tables générées à chaque mise à jour ;

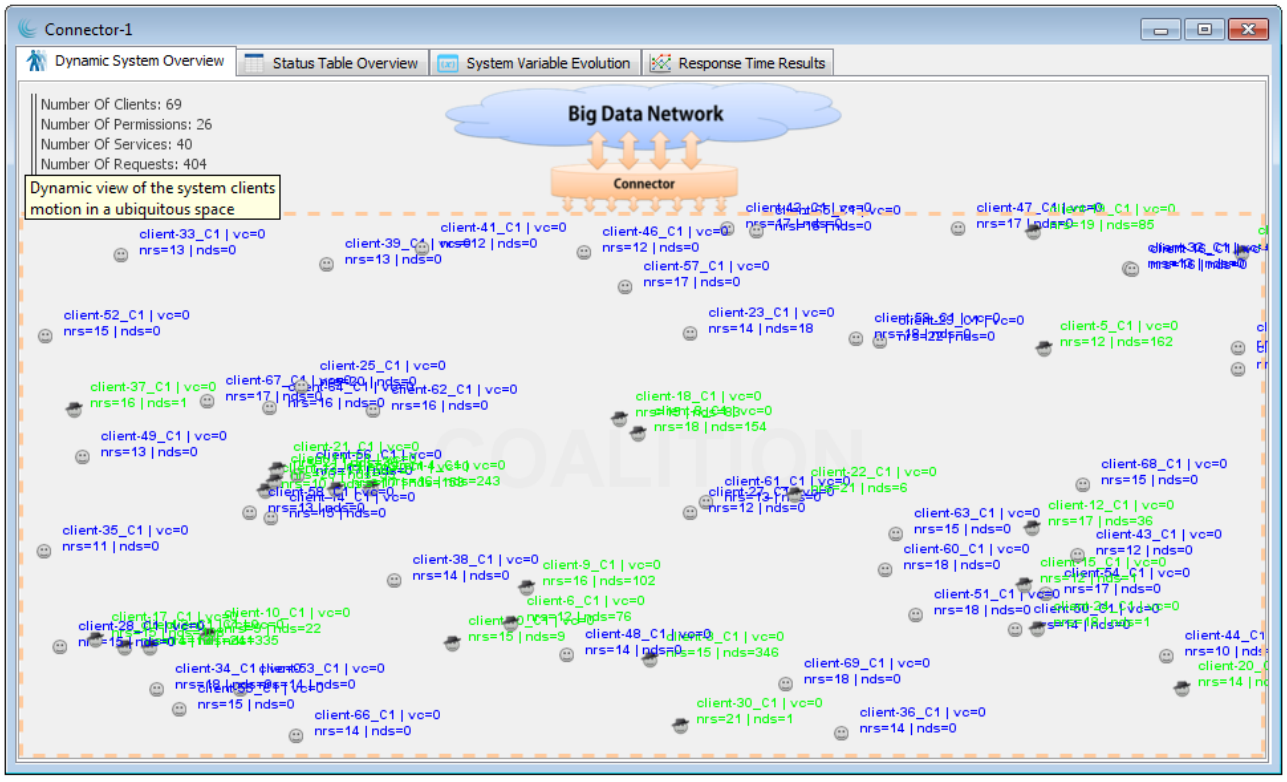



FIGURE V.4: Illustration de l'onglet de la vue dynamique du système.

– *Onglet 3 : Évolution des variables du système ("System Variable Evolution")*. Cet onglet permet d'afficher l'évolution en temps réel des paramètres du système : indices de popularité des services proposés, indices de niveau de redondance, nombre de clients actifs par service (NAC), etc. ;

– *Onglet 4 : Affichage des résultats ("Response Time Results").* Cet onglet permet de visualiser les résultats des scénarios de simulation. Il permet d'afficher les statistiques des services demandés et fournis ainsi que les temps de réponse de chaque service. La Fig. V.5 illustre de l'onglet de reporting des résultats de simulation.

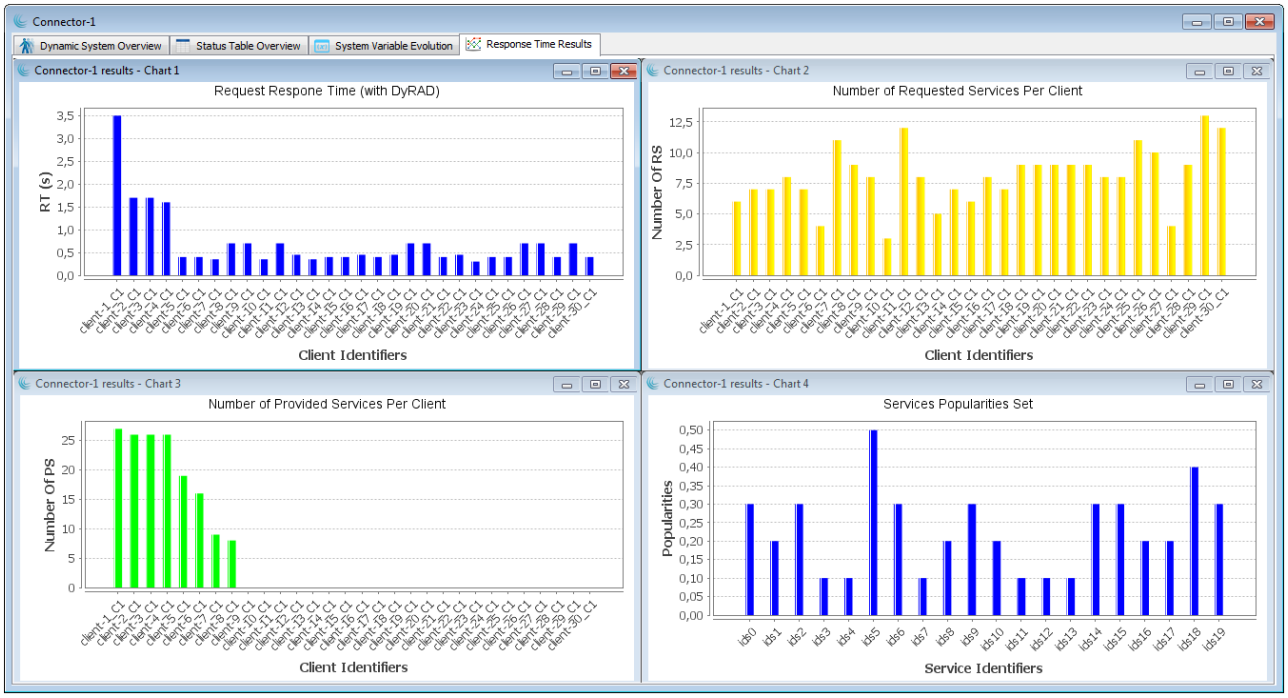



FIGURE V.5: Illustration de l'onglet de reporting.

Comme expliqué dans ce qui précède, les communications entre les agents du système (à l'aide des messages FIPA-ACL) peuvent être visualisés à partir de l'interface de configuration en démarrant l'outil SNIFFER. La Fig. V.6 illustre un exemple des communications inter-agents. Dans la suite nous présentons les principaux outils de configuration des scénarios de simulation, en particulier la configuration du connecteur et du voisinage collaboratif (ou coalition).

### V.5.2 Outils et Configurations

MASiM dispose d'un ensemble d'outils de configuration des scénarios de simulation que nous décrivons dans ce paragraphe. La configuration porte sur trois éléments :

– *le connecteur* : définition des bases de données (durées de validité des services, indices de popularité, estimations des temps de réponse du connecteur, tailles et coûts des services, etc.). La Fig. V.7 (a) illustre l'onglet de configuration du connecteur ;

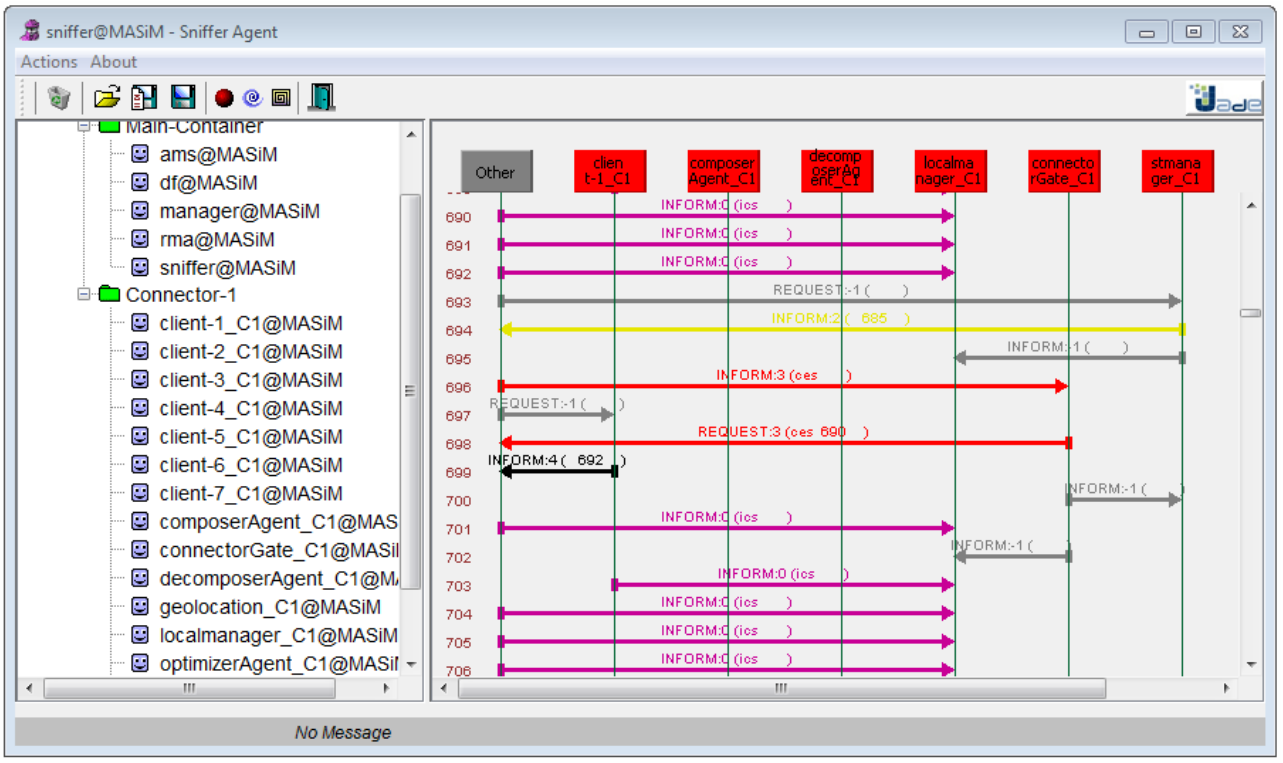



FIGURE V.6: Illustration des communications inter-agents à l'aide de l'outil SNIFFER.

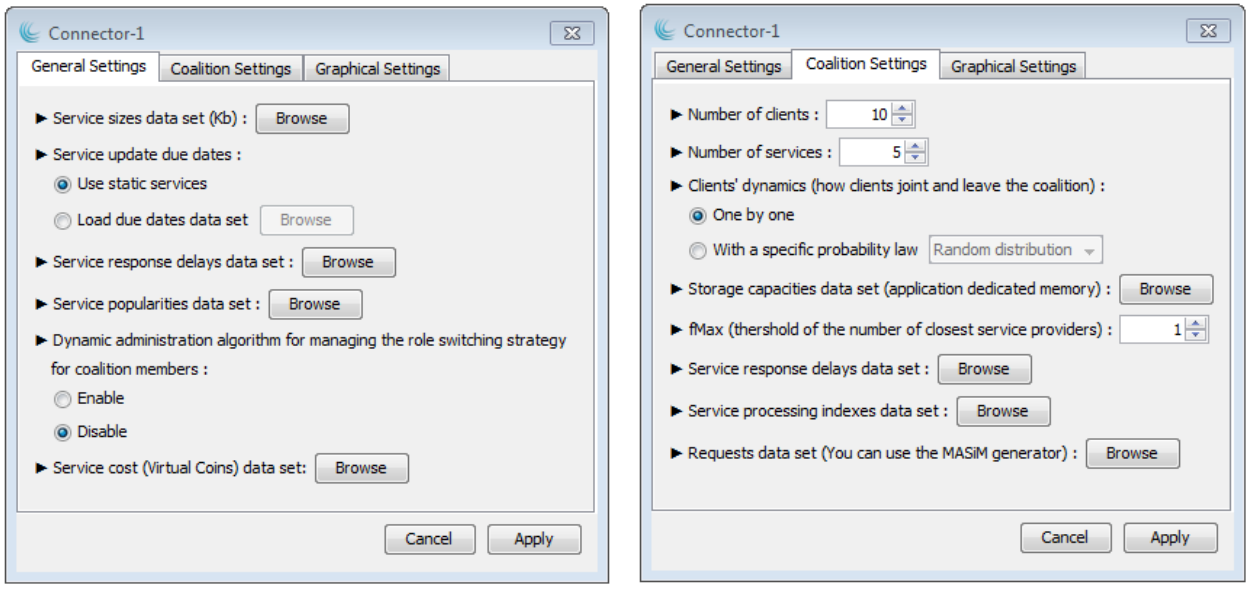
– *le voisinage collaboratif* : définition du nombre des clients, nombres de services, bases de données des requêtes clients, rayon du voisinage, etc. La Fig. V.7 (b) illustre l'onglet de configuration du voisinage collaboratif ;
– *l'affichage des données dans la fenêtre de supervision* : choix des informations à afficher (ex. nombre de services demandés, nombre de services fournis, coordonnées initiales des agents, etc.

MASiM dispose aussi d'autres outils permettant de générer d'une manière aléatoire et automatisée des bases de données d'indices de popularité et de requêtes utilisateurs. La Fig. V.8 illustre ces deux outils.

### V.5.3 Limitations et Perspectives

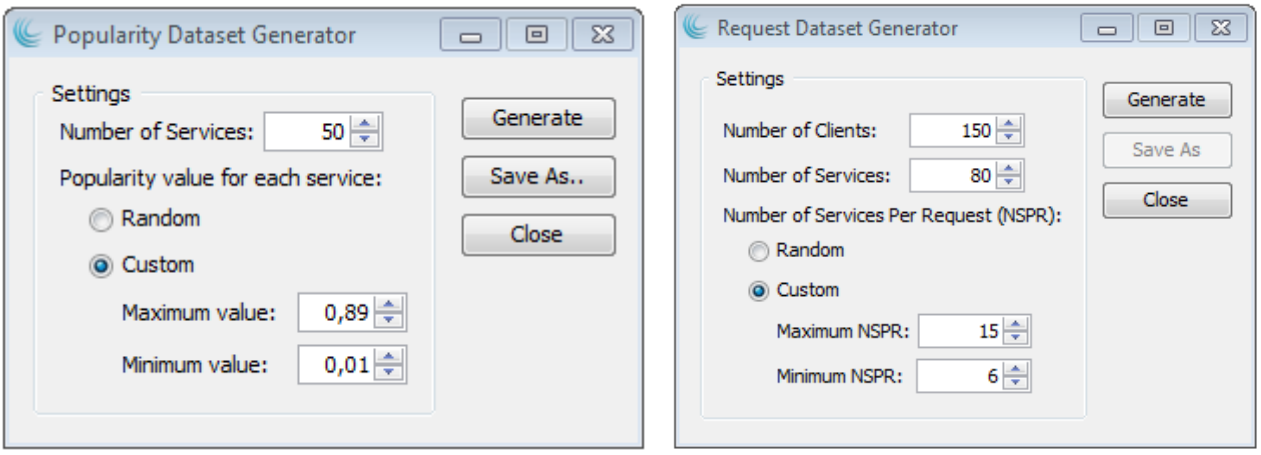
L'outil MASiM nous a permis de simuler plusieurs scénarios et nous a aidé en particulier à faire la comparaison entre plusieurs hypothèses (présentées dans la suite). Néanmoins, il existe encore des axes d'amélioration de cet outil. En effet, les limitations que présente MASiM sont les suivantes :

– La généralisation : l'outil est développé dans le contexte de cette thèse et a pour objectif la validation des approches proposées. Donc il sera un peu difficile (et non pas impossible) de l'utiliser en dehors du contexte de la thèse.




(a) Onglet de configuration du connecteur (b) Onglet de configuration du voisinage collaboratif

FIGURE V.7: Illustration des onglets de configuration du connecteur et du voisinage collaboratif.




(a) Générateur des indices de popularité (b) Générateur des requêtes utilisateurs

FIGURE V.8: Illustration des outils de génération aléatoire des indices de popularité et des requêtes utilisateurs

– L'abstraction : les classes et les méthodes développées sont aussi influencées par les algorithmes spécifiques aux travaux de la thèse. Par conséquent, la réutilisation des classes et des méthodes (entières) peut être difficile. Néanmoins, certains éléments du code présentent un haut niveau d'abstraction et peuvent être réutilisés avec facilité.

Au niveau de la généralisation de l'outil MASiM, nous pensons qu'une étude portant sur les besoins en termes d'outils de simulation des chercheurs dans le croisement des domaines de systèmes d'information, de systèmes multi-agents et d'informatique ubiquitaire, doit être menée afin d'identifier les paramètres et les outils clés d'un simulateur de grande envergure. Au niveau de l'abstraction,

nous pensons que les classes et les méthodes implémentées dans MASiM peuvent être développées en indépendance du contexte de la thèse et classées sous forme de librairies qui peuvent être utilisées avec des appels externes. Ceci réduira certainement la corrélation entre MASiM et le contexte de la thèse mais en parallèle demandera plus de travail d'adaptation pour chaque future application.

Dans le suite, nous présentons quelques scénarios de simulations afin de soutenir les approches proposées dans ce travail de thèse. Nous présentons une analyse détaillée des résultats qui démontrent l'efficacité du système proposé.

## V.6 Scénarios de Simulation et Analyse des Résultats

### V.6.1 Mise en Scène

Dans cette partie nous proposons des exemples de services pouvant être demandés par les utilisateurs dans un espace ubiquitaire ; et de réponses pouvant être délivrées par le système. En effet, nous avons défini au début (§III.2) comme étant une fonctionnalité ou une partie d'une fonctionnalité fournie par un composant logiciel (fournisseur de service) afin d'accomplir une tâche bien déterminée. En particulier, un service d'aide à la mobilité offre des informations ou une partie d'information en lien avec la mobilité des utilisateurs. Tout au long du rapport nous faisons référence à un service e utilisant son identifiant. En effet, de point de vue programmation, cet identifiant peut être généré en utilisant une fonction de hachage. Le nom du service par exemple, peut être utilisé pour créer l'identifiant et former une paire de clé, valeur. La clé correspond dans ce cas à l'identifiant du service et la valeur à son nom par exemple – et utilisé d'une manière plus flexible, efficace et concise.

A l'instant $t$ = 07h30 et pendant une période d'acquisition de 3 secondes $Ta = 3$, nous supposons qu'il existe un nombre d'utilisateurs connectés au système formulant un ensemble de requêtes. Soit l'ensemble de requêtes suivant :

- requête 1 := voyager à l'instant t de l'endroit B à l'endroit C ;
- requête 2 := voyager le prochain weekend de l'endroit A à l'endroit B avec un coût minimum, demander les prévisions météorologiques et les événements culturels pour le prochain weekend à l'endroit B ;
- requête 3 := voyager à l'instant t de l'endroit A à l'endroit C ;
- requête 4 := voyager le prochain weekend de l'endroit C à l'endroit B et demander la liste des évènements à l'endroit B ;

- requête 5 := chercher le meilleur service de liaison avec le train T, prévu à l'endroit A à $t$ = 09h00 aujourd'hui pour aller à l'endroit C ;
- requête 6 := voyager à l'instant t de l'endroit A à l'endroit B ;
- requête 7 := chercher un bon hôtel rapport qualité/prix à l'endroit D pendant le prochain weekend et faire la réservation, chercher le meilleur chemin et heure de départ pour aller de l'endroit B à l'endroit D avec la voiture, selon le trafic routier ;
- etc.

Les requêtes reçues par l'agent *Ad* sont décomposées en plusieurs demandes de services indépendantes. Le processus de décomposition des requêtes en demandes de services indépendantes (ex. en utilisant un moteur d'inférence) n'est pas détaillé dans le cadre de cette thèse car il représente en soi un axe de recherche. Ainsi, nous pouvons supposer le résultat de décomposition est le suivant :

- $s_{13}$ := chercher le plus court chemin pour aller de l'endroit C à l'endroit B le prochain weekend
- $s_{11}$ := perturbations du transport routier de l'endroit B à l'endroit C à l'instant t ;
- $s_{32}$ := prévisions météorologiques à l'endroit B le prochain weekend ;
- $s_2$ := chercher un bon hôtel qualité/prix à l'endroit D durant le weekend prochain et faire la réservation ;
- $s_{16}$ := chercher le plus court chemin pour aller avec la voiture de l'endroit B à l'endroit D ;
- $s_{28}$ := chercher la meilleure heure de départ pour aller de l'endroit B à l'endroit D avec la voiture selon les prévisions du trafic routier pendant le prochain weekend ;
- $s_{14}$ := évènements culturels à l'endroit B le prochain weekend ;
- $s_6$ := voyager de l'endroit B à l'endroit C à l'instant t ;
- $s_9$ := voyager de l'endroit A à l'endroit B le prochain weekend avec le meilleur prix/meilleur service de connexion avec le train T à 9h00 aujourd'hui) ; etc.

Le système proposé dans cette thèse permet d'optimiser la recherche des réponses des requêtes utilisateurs et la localisation des fournisseurs de services dans le RDTC. Le flux d'informations en circulation dans le réseau et en particulier les requêtes utilisateurs est le sujet d'optimisation de cette thèse. Dans ce qui suit, nous présentons des scénarios de simulation qui démontrent la capacité du système à gérer la charge et la lisser d'une manière dynamique grâce à la coopération du connecteur et du voisinage collaboratif.

Le premier scénario porte sur un exemple de génération des itinéraires des agents ramasseurs au sein d'un connecteur. Le deuxième scénario porte sur l'argumentation de l'avantage de l'approche de changement dynamique du rôle des agents utilisateurs. Le troisième scénario s'intéresse à l'étude des performances du protocole de communication développé. Le quatrième scénario démontre la capacité

du système à supporter la montée en charge en simulant un voisinage collaboratif composé de 1000 agents utilisateurs (d'où le nom du scénario 1K).

### V.6.2 Scénario 1 : Génération des Itinéraires de Agents *Ar*

Le problème de génération des itinéraires des agents $Ar$ se résout d'un manière distribuée en faisant intervenir plusieurs agents du connecteur. En effet, dans une première étape, les requêtes simultanées sont décomposées par un agent $Ad$ afin d'identifier les similarités et générer un table de services qui permet d'identifier les fournisseurs de services disponibles dans le Réseau Distribué de Transport Comodal (RDTC). Cette table de services est par la suite transférée à un agent $Ao$. Ce dernier est responsable d'optimiser globalement une solution selon les critères présentés dans le chapitre III. Ensuite, la solution générée est utilisée pour créer un ensemble d'itinéraires des agent $Ar$ qui sont responsables de collecter les informations demandées en se déplaçant dans le RDTC selon ces itinéraires. Dans ce scénario nous présentons le résultat de décomposition d'un ensemble de requêtes utilisateurs, la solution générée et les itinéraires créés.

Tous d'abord, nous supposons que pour un connecteur $C_1$ :
- Les membres du voisinage collaboratif formulent un ensemble de 10 requêtes simultanées à un instant $t$ (table V.3) ;
- La période d'acquisition $Ta = 2$ secondes.

La décomposition de l'ensemble des 10 requêtes génère (table V.4) :
- 54 services élémentaires (les similarités sont identifiées) demandés ;
- 19 fournisseurs de services disponibles dans le RDTC.

La solution générée affecte seulement 17 fournisseurs de services parmi les 19 identifiés aux différents services demandés. Les itinéraires générés pour les agents Ar sont illustrés dans la table V.5.

### V.6.3 Scénario 2 : Changement de Rôle

L'objectif de ce scénario est de démontrer les avantages de l'approche du changement de rôle et en particulier, l'importance de l'algorithme d'administration du changement dynamique du rôle. En effet, nous présentons en premier lieu une comparaison entre deux types de systèmes :

1. Le premier système est composé d'un connecteur et d'une coalition d'agents composée de clients passifs à rôles statiques, c'est-à-dire, les clients ne changent pas leurs rôles au cours du temps. Nous appelons ce système : SRBA ("Static Role-Based Architecture") ;

TABLE V.3: Identification des services demandés par les requêtes utilisateurs

| | $req_{1,1}$ | $req_{2,1}$ | $req_{3,1}$ | $req_{4,1}$ | $req_{5,1}$ | $req_{6,1}$ | $req_{7,1}$ | $req_{8,1}$ | $req_{9,1}$ | $req_{10,1}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $ids_5$ | × | | | | | | | | | |
| $ids_6$ | | | | | | | | | | × |
| $ids_8$ | × | | × | | | | | × | | |
| $ids_9$ | × | | × | | | | | × | | |
| $ids_{13}$ | × | | | | | | | | | |
| $ids_{15}$ | × | | × | | | | × | | | |
| $ids_{18}$ | × | | × | | | | | | | × |
| $ids_{19}$ | × | | × | | | | | | | |
| $ids_{21}$ | × | | | | × | | × | | | |
| $ids_{22}$ | × | | | | × | | × | | | |
| $ids_{23}$ | × | | | | × | | × | | | |
| $ids_{24}$ | × | | × | | × | | × | | | |
| $ids_{26}$ | × | | × | | | | × | | | |
| $ids_{27}$ | × | | × | | | | × | | | |
| $ids_{29}$ | | | × | | × | | | | | |
| $ids_{30}$ | × | | | | × | | | | | |
| $ids_{33}$ | × | | × | | × | | | | | |
| $ids_{34}$ | × | | × | | × | | | | | |
| $ids_{35}$ | × | | | | × | | | | | |
| $ids_{36}$ | × | | | | × | | | | | |
| $ids_{37}$ | × | | × | | × | | × | | | |
| $ids_{38}$ | × | | × | | × | | × | | | |
| $ids_{39}$ | × | | | | × | | × | | | |
| $ids_{41}$ | × | | | | | | × | | | |
| $ids_{42}$ | × | | | | | | × | | | |
| $ids_{46}$ | | | | × | × | × | | | | × |
| $ids_{48}$ | | | | | × | | | | | |
| $ids_{51}$ | × | | | | × | | × | | | |
| $ids_{53}$ | × | | | | | | × | | | |
| $ids_{54}$ | × | | | | | | × | | | |
| $ids_{59}$ | × | | | | × | | × | | | |
| $ids_{61}$ | × | | × | | × | | × | | | |
| $ids_{63}$ | × | | × | | × | | × | × | | |
| $ids_{64}$ | × | | | | × | | × | × | | |
| $ids_{65}$ | × | | | | × | | | | | |
| $ids_{66}$ | × | | × | | × | | | | | |
| $ids_{72}$ | × | × | | | × | | × | | | |
| $ids_{74}$ | × | × | | | | | × | | | |
| $ids_{76}$ | × | × | | | | | × | | | |
| $ids_{78}$ | × | | | | × | | × | | | |
| $ids_{79}$ | | | | | × | | | | | |
| $ids_{83}$ | × | | | | × | × | × | | | |
| $ids_{84}$ | | | × | | × | × | | | | |
| $ids_{85}$ | × | | × | | × | | × | × | | |
| $ids_{86}$ | × | | | | × | | × | × | | |
| $ids_{87}$ | × | | | | | | × | | | |

| | $req_{1,1}$ | $req_{2,1}$ | $req_{3,1}$ | $req_{4,1}$ | $req_{5,1}$ | $req_{6,1}$ | $req_{7,1}$ | $req_{8,1}$ | $req_{9,1}$ | $req_{10,1}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $ids_{88}$ | × | | × | | | | × | | | |
| $ids_{92}$ | × | × | × | | | | × | | | |
| $ids_{93}$ | × | × | | | | | × | | | |
| $ids_{94}$ | × | | | | | | × | × | | |
| $ids_{95}$ | × | | × | | | | × | × | × | |
| $ids_{96}$ | × | × | × | | | | | | | |
| $ids_{99}$ | | | | | | | | | | × |
| $ids_{100}$ | × | × | × | | | | | | × | |

2. Le deuxième système est celui proposé dans cette thèse. Il consiste aussi en un connecteur et un voisinage collaboratif composé d'un ensemble de clients actifs qui changent de rôle d'une manière dynamique. Nous appelons ce système : DRBA ("Dynamic Role-Based Architecture").

Ensuite nous illustrant les motivations du développement de l'algorithme de changement dynamique du rôle. Pour cela, nous supposons que :

– le connecteur propose un nombre total de services égal à 20 ;
– la coalition est composée d'un nombre total de clients égal à 30 ;
– les clients se connectent au système l'un après l'autre et chacun d'entre eux demande aléatoirement un ensemble de services ;
– le temps de réponse à une requête est le maximum des temps de réponse correspondants à chaque service demandé par cette requête.

Les résultats de simulation illustrés dans la Fig. V.9 montrent que les performances du système DRBA (proposé dans la thèse) dépassent celles du système SRBA. En effet, nous pouvons remarquer que les temps de réponse aux premières requêtes (1, 2, 3 et 4) du système DBRA sont très proches de ceux du système SRBA. Ceci peut être expliqué par le fait que la majorité des services ne sont pas encore diffusés au sein du voisinage collaboratif. Par la suite, les clients chercheront les services directement auprès du connecteur ce qui impliquera des performances équivalentes au système SRBA. A partir de la $5^{ème}$ requête, les clients commencent à échanger les services les uns avec les autres. Dans ce cas, les temps de réponses baissent d'une façon remarquable. La moyenne des temps de réponses dans le cas de la SRBA est équivalente à 3,5 fois la moyenne des temps de réponses dans le cas de la DBRA.

La Fig. V.10 illustre les performances du système par rapport au temps de réponse en fonction du nombre de clients actifs dans une coalition. En effet, pour chaque nombre de clients actifs, nous obtenons des performances différentes. Ceci peut être expliqué par le fait que les performances du système ne s'améliorent d'une manière considérable que lorsqu'une bonne répartition des services entre les membres de la coalition est effectuée. De plus, l'augmentation du nombre de clients actifs dans le

TABLE V.4: Table de services générée par l'agent *Ad*

| | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_7$ | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ | $F_{15}$ | $F_{16}$ | $F_{17}$ | $F_{18}$ | $F_{19}$ | $F_{20}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $ids_5$ | * | 0 | * | * | * | * | * | * | * | 1 | * | 0 | * | 0 | * | * | * | * | * |
| $ids_6$ | * | 0 | * | * | * | * | * | * | * | * | * | 0 | 0 | * | * | * | 1 | * | * |
| $ids_8$ | * | 0 | 1 | * | 0 | * | * | * | * | 0 | * | * | * | * | * | * | * | 0 | * |
| $ids_9$ | 0 | 0 | 0 | * | 0 | 0 | * | * | * | * | * | * | 0 | 1 | 0 | * | 0 | 0 | * |
| $ids_{13}$ | * | 0 | * | * | * | 0 | * | * | 1 | * | * | 0 | 0 | * | * | 0 | * | * | 0 |
| $ids_{15}$ | * | 0 | * | * | * | 0 | * | * | * | 1 | * | * | 0 | * | * | 0 | * | * | * |
| $ids_{18}$ | 0 | * | * | * | 0 | 0 | 0 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 1 |
| $ids_{19}$ | * | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{21}$ | 0 | 0 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $ids_{22}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 1 | * |
| $ids_{23}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | * | 0 | 0 | * | 0 | 0 | 0 | 1 |
| $ids_{24}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 |
| $ids_{26}$ | 0 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | * | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{27}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | * | * |
| $ids_{29}$ | 1 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{30}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 1 | * | * |
| $ids_{33}$ | * | 0 | * | 1 | 0 | * | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 |
| $ids_{34}$ | * | 1 | * | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{35}$ | 1 | 0 | * | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{36}$ | * | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{37}$ | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 1 |
| $ids_{38}$ | * | 0 | * | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | * | 0 | 0 | 1 | 0 | 0 | 0 |
| $ids_{39}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | * |
| $ids_{41}$ | 0 | 0 | * | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $ids_{42}$ | * | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $ids_{46}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $ids_{48}$ | 0 | 1 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 |
| $ids_{51}$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{53}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | * | 0 | 0 | 0 | 0 |
| $ids_{54}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $ids_{59}$ | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{61}$ | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | * | 0 | 0 | 0 |
| $ids_{63}$ | * | 0 | * | * | * | 1 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | * | 0 | 0 |
| $ids_{64}$ | * | 1 | * | * | * | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{65}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{66}$ | 1 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 |
| $ids_{72}$ | * | 0 | * | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 | 1 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{74}$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | * | 0 | 0 | 0 | 0 |
| $ids_{76}$ | 0 | 1 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{78}$ | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{79}$ | 1 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{83}$ | 0 | 0 | 0 | 0 | * | * | 0 | 0 | 0 | 0 | 1 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{84}$ | 1 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{85}$ | * | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{86}$ | * | 0 | * | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{87}$ | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $ids_{88}$ | 1 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{92}$ | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | * | * | 0 | 0 |
| $ids_{93}$ | * | 0 | * | 0 | 1 | 0 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{94}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | * | * | * | 0 | 0 | 0 | * | 0 | 0 |
| $ids_{95}$ | 0 | 0 | 0 | 0 | * | 0 | 0 | * | 0 | * | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{96}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $ids_{99}$ | * | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $ids_{100}$ | 0 | 0 | 1 | 0 | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | * |

TABLE V.5: Itinéraires générés par l'agent *Ao*

| | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_7$ | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ | $F_{15}$ | $F_{16}$ | $F_{17}$ | $F_{18}$ | $F_{19}$ | $F_{20}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Ar_{1,1}$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| $Ar_{1,2}$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $Ar_{1,3}$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $Ar_{1,4}$ | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $Ar_{1,5}$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

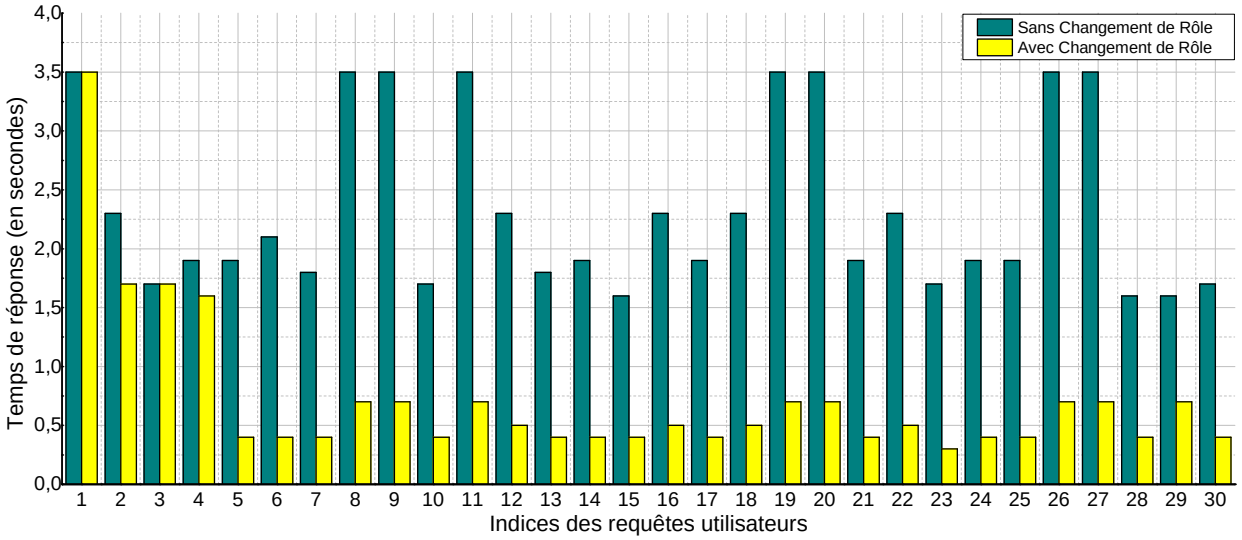



FIGURE V.9: Performance du système sans/avec changement de rôle par rapport au temps de réponse aux requêtes clients.

voisinage collaboratif n'a pas d'importance si nous ne prenons pas le critère rareté en considération. Dans ce cas, il vaut mieux d'effectuer un changement de rôle contrôlé afin d'avoir des performances nominales. En effet, plus les services demandés sont rares plus les clients qui les demande sont susceptible de devenir actifs. Il est important aussi de contrôler la redondance des services dans la coalition (un client n'est pas sensé se substituer au connecteur).

Pour démontrer l'intérêt de la stratégie d'administration dynamique du changement de rôle dans un voisinage collaboratif, nous proposons l'exemple illustré dans la table V.6. Dans cet exemple, plusieurs utilisateurs se connectent au système et composent leurs requêtes. Nous supposons que tous les utilisateurs sont autorisés à devenir des clients actifs. De cette manière nous pouvons garantir que le voisinage collaboratif contient un maximum de fournisseurs de services.

Le but de cet exemple, comme précisé précédemment, est de montrer que le nombre de clients actifs dans un voisinage collaboratif n'a pas d'importance si le système ne dispose pas d'une stratégie qui favorise la diversité des services proposés par les clients actifs afin d'optimiser les temps de réponse aux requêtes des utilisateurs.

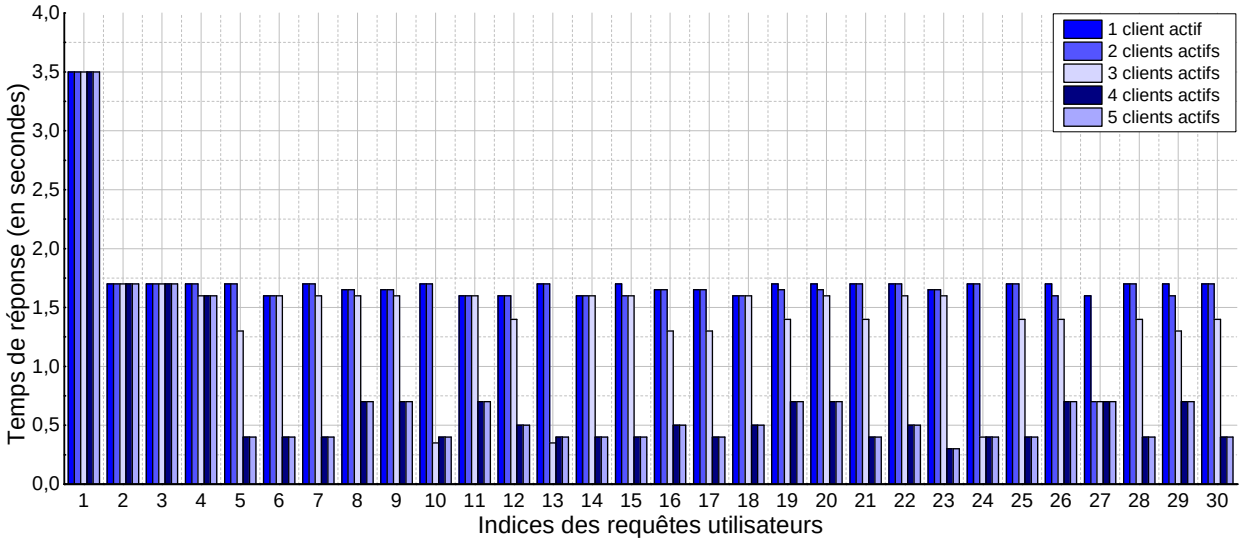



FIGURE V.10: Performance du système par rapport au temps de réponse en fonction du nombre de clients actifs dans une coalition.

Nous rappelons que le temps de réponse à une requête utilisateur dépend de l'ensemble des fournisseurs des services demandés (c'est la maximum des temps de réponse à chaque service demandé par la requête). En effet, pour l'ensemble des requêtes $req_{1,1}$, $req_{2,1}$, $req_{4,1}$, $req_{7,1}$, $req_{10,1}$, $req_{13,1}$, $req_{14,1}$ et $req_{16,1}$, le temps de réponse est influencé par les services cherchés auprès du connecteur (colorés en orange). Ce n'est qu'à partir de la requête $req_{17,1}$ (colorée en bleu) que les utilisateurs deviennent capables à échanger les informations sans l'intervention du connecteur.

En effet, ce n'est qu'à partir de la requête $req_{17,1}$ que la diversité des services au sein du voisinage collaboratif est optimale (i.e. tous les services sont disponibles au sein du voisinage collaboratif). Par conséquent, la stratégie d'administration dynamique du changement de rôle basée sur les indices de rareté et de popularité est très convenable pour assurer cette diversité et optimiser les temps de réponse au sein du voisinage collaboratif.

### V.6.4 Scénario 3 : Protocole MIPA

Dans ce scénario, nous proposons une étude comparative entre le protocole proposé MIPA et le protocole CNET. Pour cela, nous utilisons les mêmes configurations du scénario 1 (le nombre de clients, le nombre de services, le régime d'adhésion des clients au système et le calcul du temps de réponse).

L'objectif du scénario est de démontrer l'efficacité du protocole MIPA à réduire la charge de communication dans le réseau, et par conséquent, améliorer la capacité du système à supporter une charge

TABLE V.6: Exemple particulier de requêtes utilisateurs

| | $ids_1$ | $ids_2$ | $ids_3$ | $ids_4$ | $ids_5$ | $ids_6$ | $ids_7$ | $ids_8$ | $ids_9$ | $ids_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $req_{1,1}$ | × | × | × | | | | | | | |
| $req_{2,1}$ | | × | | | × | | | | | |
| $req_{3,1}$ | | | × | | × | | | | | |
| $req_{4,1}$ | × | | | | × | × | | | | |
| $req_{5,1}$ | × | | | | | × | | | | |
| $req_{6,1}$ | × | × | × | | × | | | | | |
| $req_{7,1}$ | | | × | | | | | | × | |
| $req_{8,1}$ | | | | | × | | | | × | |
| $req_{9,1}$ | | | × | | × | | | | | |
| $req_{10,1}$ | | × | | × | | | | | × | |
| $req_{11,1}$ | × | | | × | | × | | | | |
| $req_{12,1}$ | | | | × | | | | | | |
| $req_{13,1}$ | | × | | | | | | × | × | |
| $req_{14,1}$ | × | × | | | | | × | | | |
| $req_{15,1}$ | | × | × | | | | | | × | |
| $req_{16,1}$ | | | | | | | | | | × |
| $req_{17,1}$ | × | | × | | | × | | | | |
| $req_{18,1}$ | | | | | | | | | × | × |
| $req_{19,1}$ | | | | × | | × | × | × | | |
| $req_{20,1}$ | | × | | × | | | | | | × |

plus élevée. En effet, nous présentons les résultats statistiques du nombre de propositions de communication (la proposition d'une négociation génère toute une suite de messages) pour deux valeurs de rayon de voisinage $fmax$. Rappelons que le rayon du voisinage est le nombre maximal de clients actifs avec qui un agent utilisateur peut interagir lors de la demande des services au sein d'une coalition.

Comme le montrent les Fig. V.11 et V.12, le nombre de messages d'annonce établis par MIPA est particulièrement moins élevé que celui établis dans le cas de CNET. En effet, le nombre maximal des annonces dans MIPA est toujours inférieur ou égal au nombre d'annonces dans CNET. En prenant en compte la table de négociation, le nombre de messages, noté $M_{CNET}$ dans le cas de CNET (respectivement $M_{MIPA}$ dans le cas de MIPA) en utilisant chaque protocole peut être formalisé comme suit :

$$M_{CNET} = \sum_{x=1}^{W_{h,j}} Nbr(P(req_{h,j}[x])) \tag{V.1}$$

où $P(req_{h,j}[x])$ est un fournisseur (client actif) d'un service $ids_{x,j}$, $Nbr(.)$ est le nombre de tous les fournisseurs de services dans le périmètre du rayon du voisinage.

$$M_{MIPA} = \sum_{x=1}^{W_{h,j}} Nbr'(P(req_{h,j}[x])) \tag{V.2}$$

où $Nbr'(.)$ est le nombre des fournisseurs de services comptés seulement à leur première récurrence.

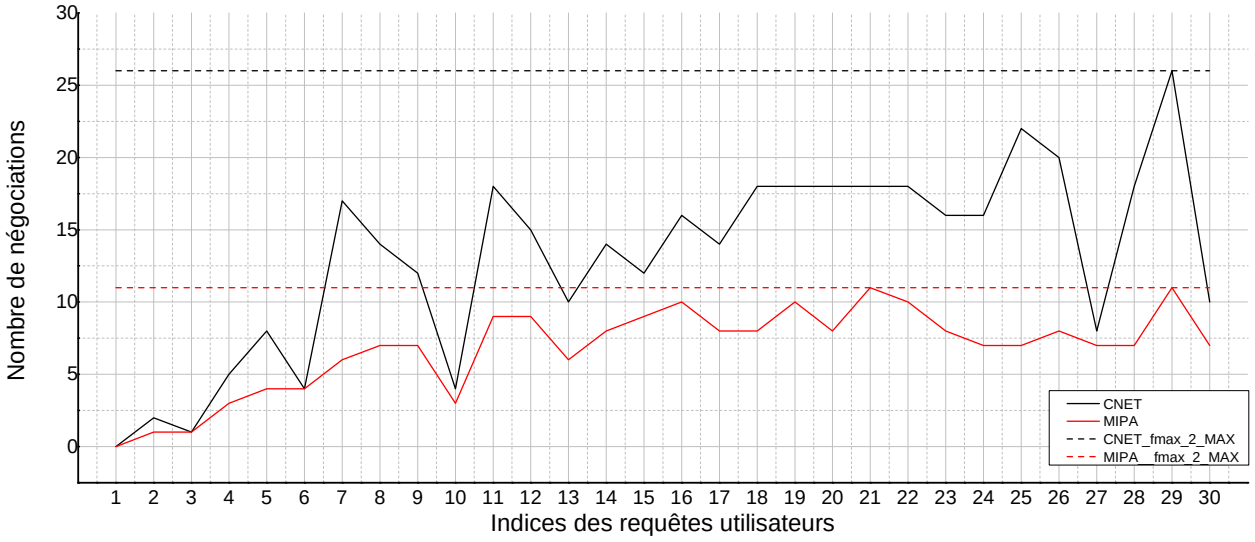



FIGURE V.11: Nombre de propositions de négociation pour un rayon de voisinage égal à 2.

Pour les deux simulations ($fmax = 2$ et $fmax = 6$, les trois premiers clients montrent un niveau très bas de propositions de négociation. La première raison peut être expliquée par le fait qu'au début, le nombre des clients actifs est très bas (les clients commencent de rejoindre le voisinage collaboratif). La deuxième raison peut être expliquée par le fait que le nombre de services demandés est très faible. Pour cela, les performances des deux protocoles peuvent paraitre similaires au départ. Cependant, du moment où le nombre de clients dans la coalition commence à augmenter, (i.e. plus de clients sont susceptibles de recevoir des permissions pour changer de rôle), les performances de MIPA commencent à dépasser celles de CNET. En résultat, nous constatons que le protocole MIPA offre une couche de communication optimisée et adaptée à notre système qui se base en une grande partie sur des terminaux mobiles à faibles ressources de stockage et de traitement.

Dans les Fig. V.11 et V.12, nous avons tracé en plus le nombre maximal de messages d'annonce (CNET_fmax_2_MAX et MIPA_fmax_2_MAX respectivement dans le cas de CNET et MIPA) issus de la simulation avec un rayon de voisinage égal à 2 (illustrée dans la Fig. V.11). Ceci permet d'observer l'augmentation en nombre de messages d'annonce qui est plus importante dans la Fig. V.12. En sachant qu'un message d'annonce correspond à la génération d'un processus de négociation, nous pouvons constater que le protocole proposé MIPA offre une meilleure performance en termes de réduction du nombre d'instances de négociation dans un voisinage collaboratif.

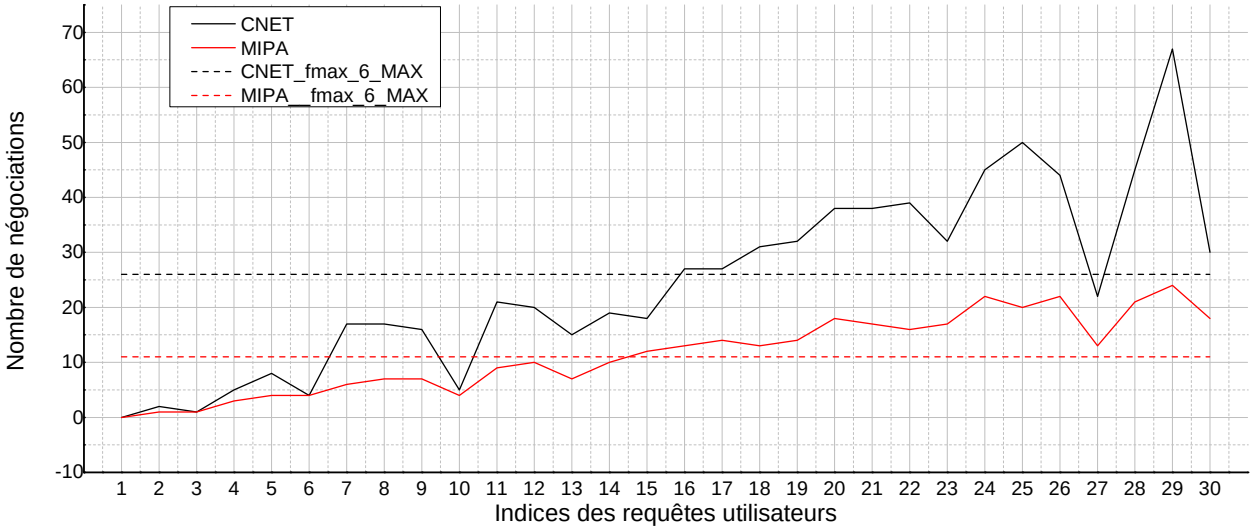



FIGURE V.12: Nombre de propositions de négociation pour un rayon de voisinage égal à 6.

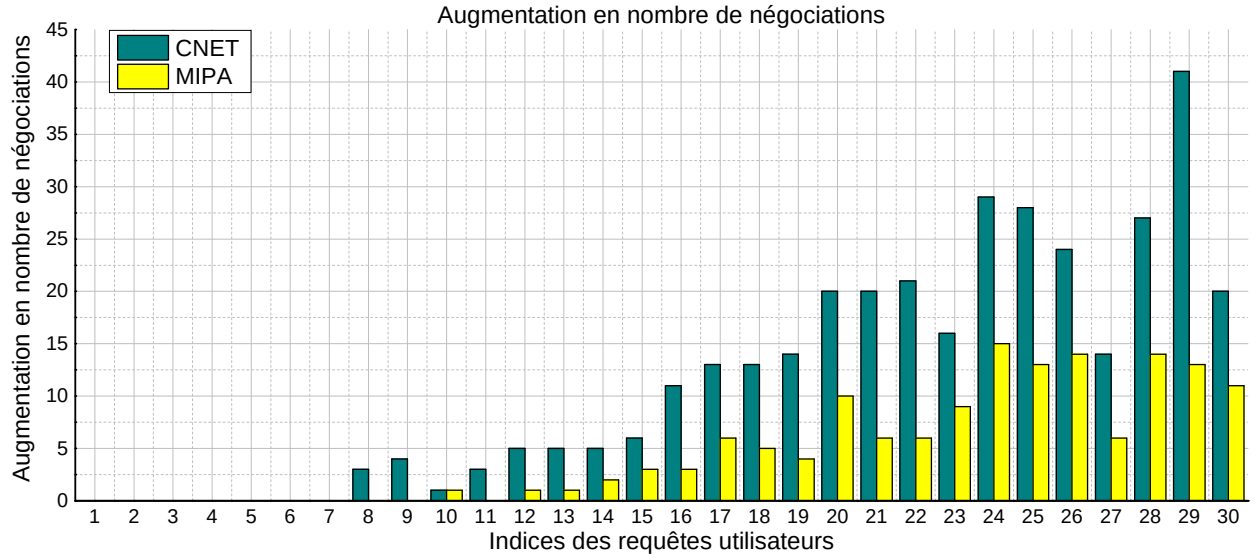



FIGURE V.13: Comparaison de l'augmentation du nombre de propositions de négociations entre MIPA et CNET.

### V.6.5 Scénario 4 : 1K

Le scénario 3 consiste à tester la robustesse du système et sa capacité à faire face à la montée en charge. Pour cela, nous avons défini les configurations suivantes :

- un nombre de clients dans le voisinage collaboratif égal à 1000 ;
- un nombre de services égal à 20 ;
- les services sont dynamiques et possèdent des popularités (illustrées dans la table V.7) et des durées de validité (illustrées dans la table V.8, DV = durée de validité) ;

– les clients rejoignent successivement la coalition et composent leurs requêtes ;
– le temps de réponse à une requête est déduit du maximum des temps de réponse aux services demandés par cette requête.

TABLE V.7: Indices de popularité utilisés

| **Service ID** | $ids_1$ | $ids_2$ | $ids_3$ | $ids_4$ | $ids_5$ | $ids_6$ | $ids_7$ | $ids_8$ | $ids_9$ | $ids_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| **Popularité** | 0,795 | 0,109 | 0,036 | 0,023 | 0,553 | 0,34 | 0,199 | 0,054 | 0,637 | 0,074 |

| **Service ID** | $ids_{11}$ | $ids_{12}$ | $ids_{13}$ | $ids_{14}$ | $ids_{15}$ | $ids_{16}$ | $ids_{17}$ | $ids_{18}$ | $ids_{19}$ | $ids_{20}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| **Popularité** | 0,355 | 0,343 | 0,385 | 0,091 | 0,437 | 0,424 | 0,872 | 0,039 | 0,494 | 0,233 |

TABLE V.8: Durées de validités utilisées

| **Service ID** | $ids_1$ | $ids_2$ | $ids_3$ | $ids_4$ | $ids_5$ | $ids_6$ | $ids_7$ | $ids_8$ | $ids_9$ | $ids_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| **DV (s)** | 612 | 80 | 70 | 60 | 400 | 120 | 130 | 90 | 60 | 760 |

| **Service ID** | $ids_{11}$ | $ids_{12}$ | $ids_{13}$ | $ids_{14}$ | $ids_{15}$ | $ids_{16}$ | $ids_{17}$ | $ids_{18}$ | $ids_{19}$ | $ids_{20}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| **DV (s)** | 80 | 300 | 240 | 100 | 150 | 180 | 70 | 90 | 110 | 130 |

La Fig. V.14 illustre le nombre de services demandés par chaque client.

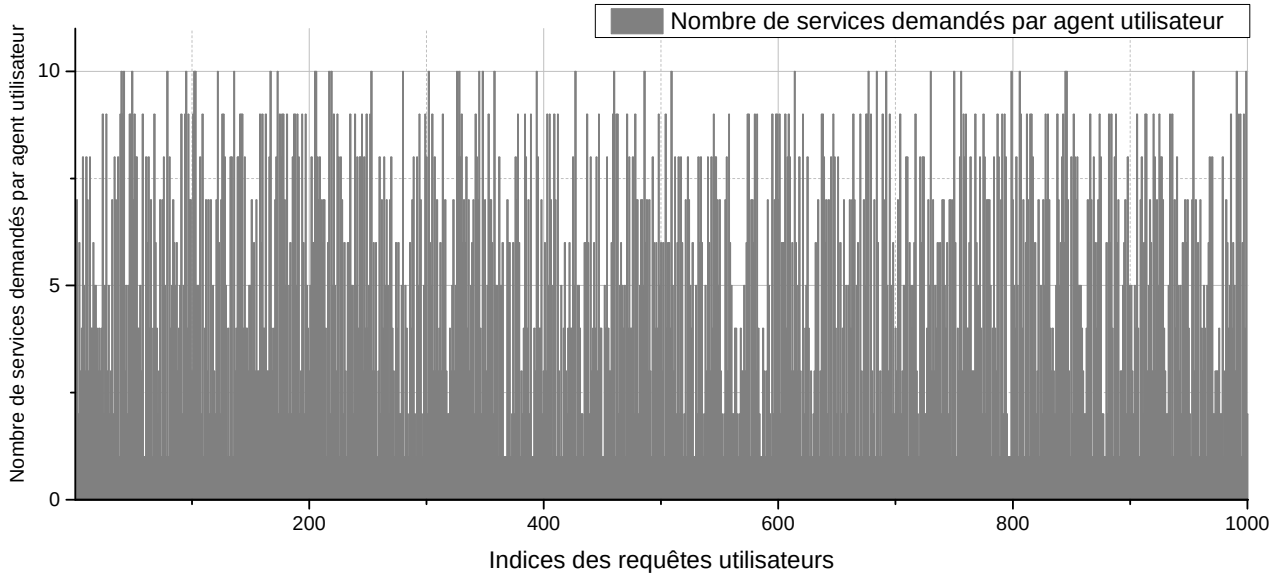



FIGURE V.14: Illustration du nombre de services demandés.

D'après la Fig. V.15, le système continue toujours à offrir les services à des temps de réponses optimisés grâce à l'approche d'administration du rôle dynamique. En dépit du nombre très élevé d'informations échangées à travers le réseau, le système continue à lisser et gérer efficacement la montée en charge. En utilisant la même configuration mais sans activer l'algorithme d'administration du changement dynamique du rôle, le système montre sa capacité à répondre à toutes les requêtes utilisateurs. Néanmoins,

les temps de réponses obtenus sont élevés. Les performances du système que nous avons proposé dans cette thèse sont meilleures que celle d'un système sans changement dynamique du rôle.

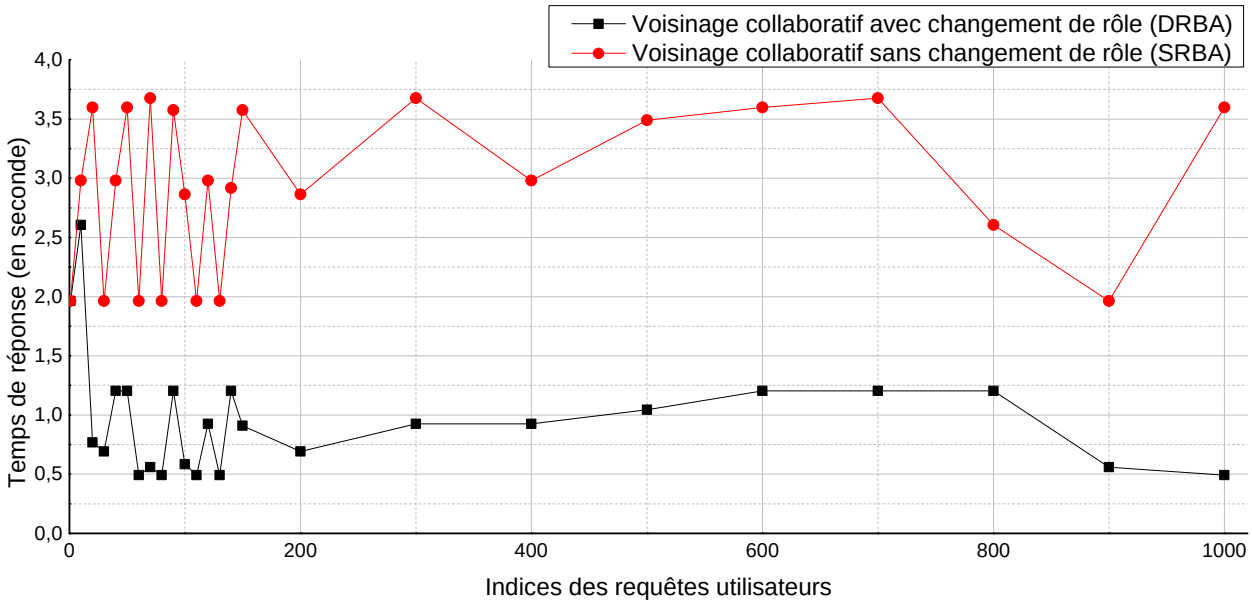



FIGURE V.15: Illustration des temps de réponse aux requêtes clients dans le cas d'un DRBA et d'un SRBA

## V.7 Conclusion

La complexité et le caractère distribué du problème de conception d'une architecture informatique d'un espace ubiquitaire nous ont amené à l'adoption d'une approche multi-agent pour la conception d'une architecture orientée-services tout en optimisant le temps de réponse ainsi que le coût des informations. Dans ce chapitre, nous avons montré le fonctionnement des approches d'optimisation adoptées et présenté les caractéristiques de la plateforme développée.

Nous avons présenté l'outil de simulation développé et nous avons illustré des scénarios de simulation réalisés à l'aide de cet outil. Les résultats de simulation prouve la robustesse du système face à un flux de données, éventuellement important, et sa capacité à s'adapter et lisser la charge du réseau. A l'aide d'une approche combinant l'optimisation distribuée et le changement dynamique du rôle, les agents utilisateurs sont capables d'obtenir une meilleure qualité de service dans le contexte d'un espace ubiquitaire.

# Conclusion Générale

## Bilan

La complexité et le caractère distribué du problème de conception d'une architecture informatique d'un espace ubiquitaire nous ont conduits vers l'adoption d'une approche multi-agent pour la conception d'une architecture orientée-services tout en optimisant le temps de réponse, le coût et la quantité de données transférée des informations de mobilité avancée. Dans cette thèse, nous avons proposé une Plateforme de Recherche et de Composition des Services d'Aide à la Mobilité (PRECSAM) capable de s'adapter au nombre élevé des utilisateurs (i.e. demandeurs de services) ainsi que de fournisseurs de services dans un espace ubiquitaire. Cette plateforme dispose d'un ensemble d'algorithmes d'optimisation qui lui permet de renforcer ses performances en termes de disponibilité, robustesse et flexibilité.

La plateforme proposée se compose de deux sous-systèmes : les connecteurs et les voisinages collaboratifs (ou coalition). En effet, un connecteur est l'entité qui se charge de la réception des demandes des utilisateurs et des tâches de recherche, de collecte et de distribution des services d'aide à la mobilité. Ces derniers sont localisés dans un Réseau Distribué de Transport Comodal (RDTC) et proposés à différents coûts, temps de réponses et quantités de données transférées. Dans ce cas, le système doit être en mesure d'optimiser la recherche des services dans le RDTC. Par conséquent, nous avons adopté une approche d'optimisation évolutionnaire basé sur des algorithmes génétiques. Par conséquent, la résolution du problème de recherche des services est donc distribuée sur les agents communicants du connecteur qui coopèrent et coordonnent leurs actions selon des cycles de traitements.

D'abord, les requêtes simultanées reçues pendant chaque cycle sont décomposées en un ensemble de services élémentaires afin d'identifier les similarités (i.e. les services demandés plusieurs fois par différents utilisateurs), affecter les services demandés aux fournisseurs de services disponibles et créer la table de services. Ensuite, cette table constitue d'entrée du processus d'optimisation qui va générer en sortie une table de services optimisée. Par la suite, cette même table est utilisée pour élaborer les itinéraires des agents ramasseurs mobiles, qui seront responsable de la collecte des données depuis le RDTC. Enfin, ces données constitueront les réponses finales du connecteur.

Les réponses générées par le connecteur sont desservies aux membres du voisinage collaboratif (i.e. les agents utilisateurs). Dotés de la capacité de changer d'une manière dynamique leurs rôles de client passif vers client actif et vice versa, les agents utilisateurs participent à l'amélioration des performances du système global. Cette capacité de changement de rôle, administrée par le connecteur, permet aux agents utilisateurs d'échanger les informations de mobilité avancée au sein d'un voisinage collaboratif. En se basant sur des indices de popularité et de redondance, le connecteur distribue les permissions de changement de rôle à certains agents utilisateurs. Ces derniers s'inscrivent dans la table d'état de la coalition en tant que fournisseurs des services dont ils sont autorisés à fournir. Chaque service proposé par les membres de la coalition possède une date de validité. A l'échéance de cette date, l'agent utilisateur proposant le service correspondant doit se désinscrire de la table d'état.

Les échanges des services au sein de la coalition sont réalisées par le moyen d'un protocole de négociation multilatérale, multicritères et à accords partiels. Ce protocole permet à un agent utilisateur de négocier un (ou plusieurs) service(s) avec plusieurs clients actifs proposant ce(s) service(s). L'originalité du protocole réside dans sa capacité à établir des accords partiels, dans le sens où un objet de négociation (i.e. un ou plusieurs services) peut être accepté ou refusé en partie. Ceci permet de mieux gérer la charge de communication dans le réseau et garantir au même temps plus grande flexibilité lors des échanges.

L'outil de simulation développé MASiM, nous a permis de générer plusieurs scénarios. Nous avons établi la comparaison entre deux types de systèmes : SRBA (Architecture basée sur un rôle statique) et DRBA (Architecture basée sur un rôle dynamique). Les résultats obtenus montrent la capacité du système à maintenir sa robustesse et réduire les temps de réponse aux requêtes des utilisateurs en se basant sur une stratégie de changement dynamique du rôle. Les résultats montrent aussi la capacité du protocole proposé à réduire la charge de communication dans le voisinage collaboratif.

# Perspectives

Parmi les développements auxquels ces travaux peuvent donner lieu, nous nous proposons de poursuivre les recherches autour des axes suivants :

Élaborer des algorithmes de gestion de la communication entre les connecteurs dispersés des différents espaces ubiquitaires afin de s'assurer de la continuité des services et de contrôler le niveau de redondance des données à l'échelle de tous les connecteurs. En effet, les utilisateurs sont des entités mobiles qui peuvent changer de position. Dans ce cas, il sera préférable de pouvoir changer de connecteur et

pouvoir continuer à fournir des services. L'existence d'une couche de gestion des services à l'échelle des connecteurs permettra de mieux répartir la charge et d'anticiper le niveau de l'offre et de la demande au sein des voisinages collaboratifs.

Étudier la possibilité d'intégrer la technique des tables de hachage distribuées (DHT [17]) pour la découverte et la localisation des services hébergés chez les agents utilisateurs dans un voisinage collaboratif. La technique utilisée dans cette thèse repose sur une table d'état distribuée à tous les membres du voisinage collaboratif. La mise à jour d'une telle table peut devenir coûteuse quand le nombre de services hébergés dans la coalition est très élevé. Les DHT permettent de répartir ces services sur les agents utilisateurs et adopter une fonction de localisation que permet d'estimer le fournisseur d'un service à partir de l'identifiant de ce dernier.

L'outil de simulation développé peut être généralisé et implémenté d'une manière plus abstraite afin de pouvoir l'utiliser pour développer des outils de simulation plus flexibles et adaptés à plusieurs scénarios et cas d'utilisation.

17. DHT : Distributed Hash Tables

# Annexe A

# Services Ubiquitaires pour le Stade Pierre Mauroy

La veille technologique effectuée dans le cadre d'une coopération entre le laboratoire LAGIS et l'entreprise ARISONA Consulting & Technology propose plusieurs services qui aident à la mobilité des personnes, assurent le divertissement et préservent la sécurité dans un territoire ubiquitaire. Dans cet annexe nous présentons un extrait de ces services.

**1. Services dans le Stade/Aréna**

Ces services comportent des services d'aide à la mobilité ainsi que des services de divertissement dédiés aux utilisateurs dans l'espace du stade/aréna.

- **Service U-Fun** : Ce service permet d'organiser des chorégraphies avant et au cours des spectacles. Il permet d'organiser des messages géants au cours des matchs en fonction des places dans les gradins. Au cours d'un concert, il est possible de faire des chorégraphies avec la lumière des écrans des Smartphones en créant des alchimies et des mouvements de lumière ainsi que le suivi des paroles des chansons.
- **Service U-Goal** : Ce service propose aux utilisateurs, lors du visionnement d'un but ou d'une occasion de but, de changer l'angle de vue en choisissant une autre caméra (pour se rassurer des décisions de l'arbitrage par exemple). Il permet aussi aux supporteurs de regarder les buts et les occasions de tous les matchs qui se jouent en même temps.
- **Service U-Player** : Ce service regroupe toutes les informations des célébrités (ex. joueurs, chanteurs, comédiens, humoristes, etc.) en fonction du spectacle en cours. Il permet d'avoir une fiche pour chaque joueur (ex. statistiques de buts, cartons, etc.) télécharger ou voir en streaming

les meilleurs buts de l'équipe ou les meilleurs vidéo-clips des musiciens, et avoir les statistiques du match en temps réel. Ce service offre également à l'utilisateur les résultats en temps réel des matchs qui se jouent en parallèles ainsi que les statistiques (Possession de balle, cartons, tirs cadrés, buts, etc.).

**2. Services de Transport**

Pour faciliter le déplacement depuis et vers le stade, les deux services suivant sont identifiés :

– **Service U-Parking** : ce service permet de montrer le nombre de places libres dans un parking, aider le conducteur à trouver sa place dans le parking et détecter leurs prix.
– **Service U-GPS+** : ce service permet aux personnes à mobilité réduite de pouvoir circuler dans le stade ou ses alentours grâce à un GPS vocal et visuel qui les guidera vers l'endroit souhaité.

**3. Services de Tourisme**

Tout au long du voyage vers le stade, l'utilisateur peut profiter des services suivants :

– **Service U-Map** : ce service est utilisé entre des amis, membres d'une famille, groupes de touristes, etc. Il permet le partage en temps réel de la position des adhérents de l'application pour faciliter leurs rencontres et donne le sentiment de la promenade en groupe.
– **Service U-Travel** : ce service permet d'acheter en ligne des tickets numériques, jouer à des jeux multi-joueurs, partager des photos, accéder à une bibliothèque en ligne (ex. livres électroniques, morceaux de musique, films), suggérer des plans touristiques et noter la qualité des services.

**4. Services de Culture**

– **Service U-Culture** : ce service permet de fournir des informations multimédia sur les monuments de la ville et leurs histoires.
– **Service U-Book** : ce service représente une bibliothèque numérique en ligne, (concept de la bibliothèque municipale), qui permet de choisir les livres selon une recherche par nom, ISBN, ou auteur par exemple. Le service enregistre les préférences des lecteurs et afin de leurs proposer d'une manière intelligente des livres pouvant les intéresser.

# Annexe B

# Fonctions de Découverte des Services dans les Systèmes Pair-à-Pair

**1. Systèmes P2P Purs**

Les systèmes P2P purs se distinguent par la décentralisation totale des données et des références de données. La fonction de découverte des services dans ce type de systèmes est essentielle pour définir la flexibilité du système et sa capacité à monter en charge.

Certains systèmes utilisent des fonctions très simples pour la recherche du contenu. Tel est le cas avec le système qui utilise la technique de l'inondation du réseau (traduction du terme anglais *flooding*) pour le routage des requêtes de découverte des services [Portmann et al., 2001] L'inondation est un algorithme simple de recherche qui consiste en deux conditions :

– Chaque nœud agit en tant que pair ; i.e. chaque nœud peut à la fois lancer et recevoir des messages de recherche de services ;
– Chaque nœud peut renvoyer un message de recherche à tous les nœuds à qui il est connecté sauf le nœud expéditeur du message.

En effet, au départ, chaque pair essaye d'établir une connexion avec au moins un autre pair. La recherche d'un service s'effectue dans ce cas en envoyant une requête à tous les pairs à qui l'utilisateur est connecté. Ensuite, chaque pair effectue une recherche locale. Dans le cas d'un succès, la recherche s'arrête et la réponse est renvoyée au pair qui a lancé la recherche. Dans le cas d'un échec, le pair effectue un renvoi (appelé "Hop") de la requête à d'autre pairs. La recherche dans ce cas s'arrête lorsqu'un nombre maximal de renvois est atteint indépendamment du succès ou de l'échec de l'opération de recherche.

D'une part, le temps de recherche est linéaire par rapport au nombre de pairs à chaque renvoi. D'une autre part, ce type de système présente un comportement polluant vis-à-vis du réseau. En effet, l'opération de découverte des services consomme énormément de bande passante vu le nombre croissant des messages à chaque recherche.

D'autres systèmes utilisent des fonctions de découverte de services plus complexes. Les protocoles P2P purs tels que Kademlia [Maymounkov et Mazières, 2002] et Chord [Stoica et al., 2001] se basent sur les tables de hachage distribuées.

Chord par exemple, est un protocole de recherche distribuée qui permet de localiser un nœud dans le réseau à partir d'une clé (une clé peut être une référence vers une ressource). D'autres applications de distribution des données peuvent être construites en se basant sur ce protocole.

Dans ce cas chaque morceau de données stocké dans la mémoire locale d'un nœud sera associé à une clé. Cette clé permet à chaque nœud du réseau peut être utilisé pour effectuer deux tâches :

– Publier des nouvelles données sur le réseau sous forme d'une paire (clé, valeur), cette opération est effectué par une méthode appelée PUT ;
– Soumettre des requêtes de recherche d'une clé, cette opération est effectué par une méthode appelée GET.

Les pairs (respectivement les morceaux de données) sont associés à une valeur de hachage nommée identifiant (respectivement clé notée $k$) composée de $m-bits$. L'identifiant d'un nœud peut être obtenu en calculant l'empreinte cryptographique (appelée aussi condensé) de son adresse IP [1]. $m$ doit être suffisamment long pour rendre négligeable la probabilité d'avoir des hachages identiques de deux pairs ou données différentes. L'ensemble des valeurs de hachage (identifiants et clés) représente l'espace des identifiants. Le tableau B.1 présente un récapitulatif des variables du système.

Table B.1: Récapitulatif des variables du système

| **Variable** | **Définition** |
|---|---|
| $n$ | Identifiant d'un noeud (valeur de hachage associée à un pair) |
| $k$ | Identifiant d'une clé (valeur de hachage associée à un morceau de données) |
| $m$ | Nombre de bits composant un identifiant |
| $N$ | Nombre des nœuds du système |
| successeur | Le prochain nœud dans l'espace des identifiants |
| prédécesseur | Le nœud précédent dans l'espace des identifiants |
| pas | $2^{i-1}$, $i \in [1, m]$ |

1. Internet Protocol : identifiant permanent ou provisoire attribué à tout objets connecté à internet. Actuellement, il existe deux type d'IP en utilisation : l'IP version 4 et l'IP version 6.

En utilisant Chord [Stoica et al., 2001, Dabek, 2005], l'espace des identifiants est représenté sous la forme d'un cercle constitué de $2^m$ nœud virtuels (des valeurs allants de 0 à $2^{m-1}$).

Tout d'abord, les identifiants des nœuds du réseau modulo $2^m$ sont dans ce cas ordonnés dans ce cercle. Ensuite, chaque clé de donnée $k$ est associée au premier nœud dont l'identifiant est égal ou supérieur à $k$. Le nœud est dans ce cas appelé successeur de cette clé et noté $successeur(k)$. Le successeur d'une clé $k$ est le premier nœud dont l'identifiant suit $k$ *modulo* $2^m$ dans le cercle des identifiants en suivant le sens horaire.

Pour chercher une clé, la requête n'a pas besoin de parcourir tous les nœuds du réseau parce que le système offre d'autres informations de routage qui ont un effet accélérateur. Ces informations d'accélération sont regroupées dans une table appelée "finger table". Pour une clé de $m - bits$, la "finger table" d'un nœud $n$ est composée de $m$ lignes. La i$^{\text{ème}}$ ligne de la table contient l'identifiant du premier nœud $s$ qui suit $n$ d'au moins $2^{i-1}$ pas dans le cercle des identifiants :

$$s = successeur(n + 2^{i-1})_{i \in [1,m]} \tag{B.1}$$

Le nœud $s$ est appelé le i$^{\text{ème}}$ doigt du nœud $n$. De cette façon, Chord [Stoica et al., 2001, Dabek, 2005] peut garantir un temps de recherche dans le pire des cas de l'ordre de $O(\log N)$, avec $N$ le nombre des nœuds du système. La Fig. B.1 présente un exemple de la topologie du réseau avec Chord, avec $m = 5$ et $N = 9$.

Les réseaux P2P purs peuvent être classifiés en deux catégories : les réseaux structurés (ex. Chord [Stoica et al., 2001] et Kademlia [Maymounkov et Mazières, 2002]) et les réseaux non structurés (ex. [Portmann et al., 2001]). [Lua et al., 2005] et [Alima et al., 2005] présentent plus de détails sur la classification des systèmes P2P selon leurs structures.

Les systèmes P2P purs donnent des poids équivalents à chaque nœud du système sans prendre en considération son niveau de performance (i.e. capacité de calcul et de stockage, bande passante, disponibilité). Ceci peut mener à la réduction des performances globales du système au fur et à mesure que des nœuds moins performants se connectent. Les systèmes P2P purs présentent un niveau très faible de gestion du moment où chaque pair est le gestionnaire de lui même. Certains systèmes P2P optent pour une découverte aveugle des services basée sur l'inondation du réseau par les requêtes de recherche et de routage, ce qui limite la montée en charge. Dans le paragraphe suivant nous présentons la catégorie des systèmes P2P hybrides.

**2. Systèmes Pair-à-Pair Hybrides**

Les systèmes P2P hybrides se distinguent par la centralisation des métadonnées (i.e. données

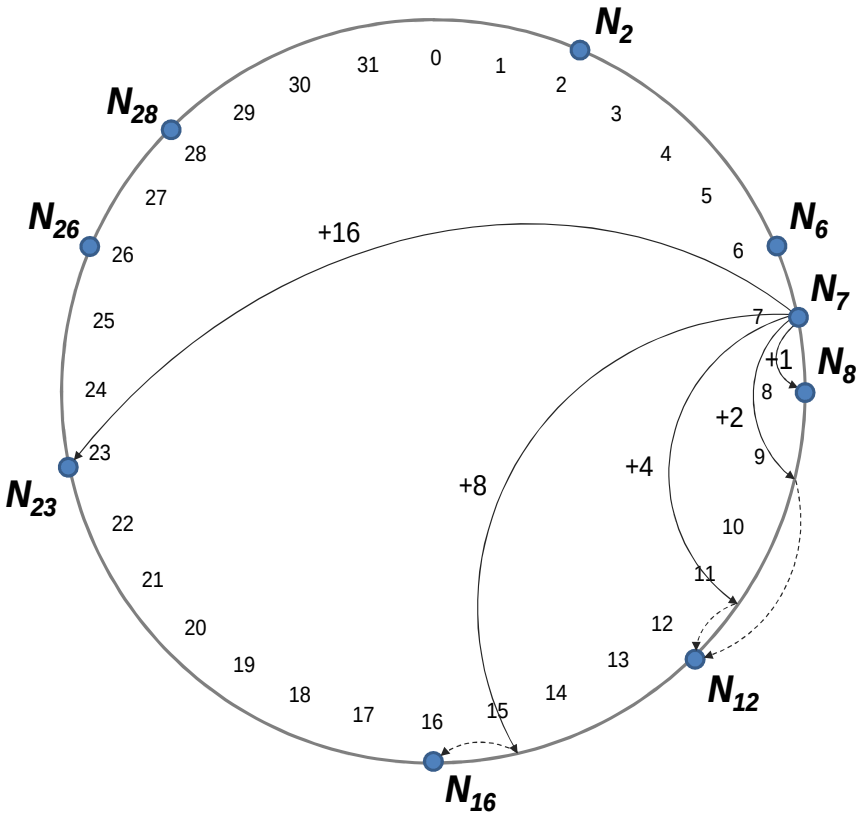



FIGURE B.1: Topologie du réseau avec Chord - Le successeur du noeud $N_7$ à 1 pas est le noeud $N_8$, à 2 et 4 pas est le noeud $N_{12}$, etc.

d'indexations) et la décentralisation des données. Ce type de systèmes regroupe les éléments d'un système P2P pur et d'un système client/serveur classique. Dans ce cas, la fonction de découverte des services consiste en une simple consultation d'un serveur de fichiers de références. Les systèmes P2P hybrides ont l'avantage d'être plus facile à gérer et possèdent un large spectre de domaines d'application. D'une part, ils offrent un meilleur contrôle des données échangées en termes de sécurité et de droit d'auteur. D'autre part, la performance du système en termes de recherche et localisation des informations est meilleure par rapport à un système P2P pur.

Par ailleurs, plusieurs systèmes P2P hybrides ont été proposés. Cependant, quelques uns ont pu réussir le test de robustesse concernant l'utilisation journalière intensive d'une large communauté d'utilisateurs. Le système de partage des fichiers BitTorrent [Zhang et al., 2010, Zhang et al., 2009] est l'un de ces systèmes. Selon [Pouwelse et al., 2005], le trafic BitTorrent représente 10 à 20% du trafic Internet mondial.

En effet, BitTorrent est un protocole de téléchargement des fichiers entre des utilisateurs privés. L'ensemble des clients participant à la tâche de distribution d'un fichier donné sont appelé "swarm" (essaim en français). Les fichiers échangés sont divisés en milliers de morceaux [Han et al., 2014]. Les

utilisateurs intéressés par un fichier donné téléchargent les morceaux qui composent ce fichier et en même temps les transmettent à d'autres utilisateurs afin d'éviter le comportement "égoïste". Un client qui télécharge et transmet des morceaux de données en même temps est appelé "leecher". Chaque pair est responsable de maximiser son ratio de téléchargement en se connectant à des pairs convenables. Les pairs qui possèdent un taux de téléchargement ascendant très élevé seront capables de télécharger avec une vitesse élevée. Finalement, lorsqu'un pair finit le téléchargement d'un fichier, il peut devenir un "seed" (fournisseur) en restant connecté et en offrant un partage gratuit du fichier.

Pour trouver un fichier avec BitTorrent, les utilisateurs accèdent à des sites web spéciaux. Ces sites web sont en effet des annuaires globaux des fichiers disponibles. Il suffit de télécharger un fichier de métadonnées référençant le contenu cherché et l'exécuter avec le client BitTorrent. Le nouveau contenu est injecté manuellement dans le système à travers des modérateurs qui éliminent les fichiers truqués, illégaux, de très mauvaise qualité ou avec une fausse dénomination.

En se basant sur la définition proposée précédemment, le système BitTorrent est P2P parce que les nœuds du système demandent des services aux autres nœuds (i.e. téléchargement des morceaux de données) et offrent des services (i.e. transfert des données) en même temps.

Le système de partage des fichiers musicaux Napster est un autre exemple de système de distribution de contenue basé sur une architecture P2P hybride [Yang et Garcia-Molina, 2001]. Dans Napster, un ou plusieurs serveurs centraux sont responsables de l'indexation des fichiers musicaux dans le système. La recherche des données s'exécute sur l'un de ces serveurs. Une fois le fichier est trouvé, l'utilisateur le récupère directement du pair qui détient le fichier.

Les système P2P hybrides traitent efficacement le problème de gestion des nœuds du réseau et assurent un contrôle global des informations de découverte de service et de routage. Néanmoins, la centralisation de l'indexation des données peut mener à un goulot d'étranglement et donc à l'échec total du système. Dans ce cas, des approches de distribution de l'indexation des données ainsi que la redondance des serveurs d'indexation peuvent être prises en considération afin de maximiser les performances du système et améliorer sa robustesse vie-à-vie de la monter en charge.

Dans une architectures centralisée (i.e. basée sur un modèle client/serveur), les RTS du système sont concentrées dans un seul serveur central ou en une grappe à petit nombre de serveurs qui sera responsable de la gestion des flux de communication (i.e. requêtes d'inscription, routage, recherche des services, demande et réponse, etc.) ainsi que la régulation de la charge dans le réseau. La centralisation de l'accès à l'information dans les systèmes fortement sollicités augmente énormément la probabilité

d'un échec total surtout lorsque l'environnement présente des limitations par rapport à la bande passante dans le réseau (i.e. création des goulots d'étranglement). Ces problèmes ont motivé les chercheurs à inventer des nouvelles approches afin de distribuer la charge de traitement et la bande passante sur les différents nœuds du système distribué. Les systèmes P2P font partie de cette invention qui offre une alternative au modèle c/s traditionnel et qui ont démontré leur capacité à monter en charge tout en gardant des temps de réponse acceptables.

# Annexe C

# Coordonnées des Agents Utilisateurs

TABLE C.1: Coordonnées de agents utilisateurs de l'exemple du paragraphe IV.4.2.

| | **Coordonnées** | |
|---|---|---|
| **Identifiants** | **X** | **Y** |
| $a_{1,1}$ | 10 | 10 |
| $a_{2,1}$ | 90 | 90 |
| $a_{3,1}$ | 20 | 5 |
| $a_{4,1}$ | 80 | 80 |
| $a_{5,1}$ | 30 | 67 |
| $a_{6,1}$ | 70 | 18 |
| $a_{7,1}$ | 40 | 44 |
| $a_{8,1}$ | 60 | 66 |
| $a_{9,1}$ | 50 | 25 |
| $a_{10,1}$ | 50 | 11 |
| $a_{11,1}$ | 60 | 30 |
| $a_{12,1}$ | 40 | 10 |
| $a_{13,1}$ | 50 | 50 |
| $a_{14,1}$ | 30 | 90 |
| $a_{15,1}$ | 40 | 82 |
| $a_{16,1}$ | 20 | 62 |
| $a_{17,1}$ | 30 | 30 |
| $a_{18,1}$ | 10 | 25 |
| $a_{19,1}$ | 20 | 45 |

| | **Coordonnées** | |
|---|---|---|
| **Identifiants** | **X** | **Y** |
| $a_{20,1}$ | 90 | 45 |
| $a_{21,1}$ | 10 | 76 |
| $a_{22,1}$ | 80 | 55 |
| $a_{23,1}$ | 20 | 18 |
| $a_{24,1}$ | 70 | 45 |
| $a_{25,1}$ | 30 | 5 |
| $a_{26,1}$ | 60 | 95 |
| $a_{27,1}$ | 40 | 25 |
| $a_{28,1}$ | 50 | 75 |
| $a_{29,1}$ | 50 | 35 |
| $a_{30,1}$ | 40 | 60 |
| $a_{31,1}$ | 60 | 8 |
| $a_{32,1}$ | 30 | 10 |
| $a_{33,1}$ | 70 | 65 |
| $a_{34,1}$ | 20 | 31 |
| $a_{35,1}$ | 80 | 2 |
| $a_{36,1}$ | 5 | 15 |
| $a_{37,1}$ | 95 | 25 |
| $a_{38,1}$ | 14 | 14 |

# Annexe D

# Extrait de la Console de MASiM

```
---------------------------------
> Lanching C1
> Retrieving CommandDispatcher for platform null
> Lanching C2
> Retrieving CommandDispatcher for platform null
> connectorGate_C1:/> Starting..
> geolocation_C1:/> Starting..
> stmanager_C1:/> Starting..
> connectorGate_C2:/> Starting..
> geolocation_C2:/> Starting..
> stmanager_C2:/> Starting..
> client-1_C2:/> Starting..
> client-1_C2:/> request = [1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1]
> client-1_C2:/> updating my status table..
> client-1_C2:/> starting the identification process..
> client-1_C2:/> processing my request to find suitable service providers..
> connectorGate_C2:/> processing a request received from: client-1_C2
> connectorGate_C2:/> sending a permission to: client-1_C2
> connectorGate_C2:/> updating the status table
> client-1_C2:/> 3.5
> client-2_C2:/> Starting..
> client-2_C2:/> request = [1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
```

```
0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0]
> client-2_C2:/> updating my status table..
> client-2_C2:/> starting the identification process..
> client-2_C2:/> processing my request to find suitable service providers..
> connectorGate_C2:/> processing a request received from: client-2_C2
> client-1_C2:/> processing a request received from: client-2_C2
> connectorGate_C2:/> sending a permission to: client-2_C2
> client-2_C2:/> 2.1
> connectorGate_C2:/> updating the status table
> client-1_C1:/> Starting..
> client-1_C1:/> request = [1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0]
> client-1_C1:/> updating my status table..
> client-1_C1:/> starting the identification process..
> client-1_C1:/> processing my request to find suitable service providers..
> connectorGate_C1:/> processing a request received from: client-1_C1
> client-1_C1:/> 3.5
> client-1_C2:/> The service ids38 is no more provided..
> client-3_C2:/> Starting..
> client-3_C2:/> request = [0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1]
> client-3_C2:/> updating my status table..
> client-3_C2:/> starting the identification process..
> client-3_C2:/> processing my request to find suitable service providers..
> connectorGate_C2:/> processing a request received from: client-3_C2
> connectorGate_C2:/> sending a permission to: client-3_C2
> connectorGate_C2:/> updating the status table
> client-1_C2:/> processing a request received from: client-3_C2
> client-2_C2:/> processing a request received from: client-3_C2
> client-3_C2:/> 1.7
> client-4_C2:/> Starting..
> client-4_C2:/> request = [0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0,
0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1]
> client-4_C2:/> updating my status table..
```

```
> client-4_C2:/> starting the identification process..
> client-4_C2:/> processing my request to find suitable service providers..
> connectorGate_C2:/> processing a request received from: client-4_C2
> connectorGate_C2:/> sending a permission to: client-4_C2
> connectorGate_C2:/> updating the status table
> client-2_C2:/> processing a request received from: client-4_C2
> client-1_C2:/> processing a request received from: client-4_C2
> client-3_C2:/> processing a request received from: client-4_C2
> client-4_C2:/> 1.6
> client-2_C1:/> Starting..
> client-2_C1:/> request = [1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
0, 1, 1]
> client-2_C1:/> updating my status table..
> client-2_C1:/> starting the identification process..
> client-2_C1:/> processing my request to find suitable service providers..
> connectorGate_C1:/> processing a request received from: client-2_C1
> client-2_C1:/> 2.25
> client-3_C2:/> The service ids38 is no more provided..
> client-5_C2:/> Starting..
> client-5_C2:/> request = [1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1,
0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
> client-5_C2:/> updating my status table..
> client-5_C2:/> starting the identification process..
> client-5_C2:/> processing my request to find suitable service providers..
> connectorGate_C2:/> sending a permission to: client-5_C2
> connectorGate_C2:/> updating the status table
> client-4_C2:/> processing a request received from: client-5_C2
> client-2_C2:/> processing a request received from: client-5_C2
---------------------------------
```

# Bibliographie

[Abdo et al., 2014] Abdo, J., Demerjian, J., Chaouchi, H., Barbar, K., et Pujolle, G. (2014). Single-sign-on in operator centric mobile cloud architecture. In *Mediterranean Electrotechnical Conference (MELECON), 2014 17th IEEE*, pages 151–155.

[Adam et Mandiau, 2007] Adam, E. et Mandiau, R. (2007). Flexible roles in a holonic multi-agent system. In *Holonic and Multi-Agent Systems for Manufacturing, Third International Conference on Industrial Applications of Holonic and Multi-Agent Systems, HoloMAS 2007, Regensburg, Germany, September 3-5, 2007, Proceedings*, pages 59–70.

[Alima et al., 2005] Alima, L., Ghodsi, A., et Haridi, S. (2005). A framework for structured peer-to-peer overlay networks. In *Global Computing*, volume 3267, pages 223–249. Springer Berlin Heidelberg.

[Alshareef et Grigoras, 2014] Alshareef, H. et Grigoras, D. (2014). Mobile ad-hoc network management in the cloud. In *Parallel and Distributed Computing (ISPDC), 2014 IEEE 13th International Symposium on*, pages 140–147.

[Anderson, 2004] Anderson, D. P. (2004). Boinc : A system for public-resource computing and storage. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, GRID'04, pages 4–10.

[Anderson et al., 2002] Anderson, D. P., Cobb, J., Korpela, E., Lebofsky, M., et Werthimer, D. (2002). Seti@home : An experiment in public-resource computing. *Commun. ACM*, 45(11) :56–61.

[André Filipe de Moraes et al., 2011] André Filipe de Moraes, B., Maria das Graças Bruno, M., Wagner Tanaka, B., Guiou, K., Brunno dos Passos, A., de Castro, S., et Terry Lima, R. (2011). *Principles of Agent-Oriented Programming.* InTech.

[Armbrust et al., 2010] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et Zaharia, M. (2010). A view of cloud computing. *Commun. ACM*, 53(4) :50–58.

[Armbrust et al., 2009] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., et Zaharia, M. (2009). *Above the Clouds : A Berkeley View of Cloud Computing.*

[Armoogum et al., 2010] Armoogum, J., Bouffard-Savary, E., Caenen, Y., Couderc, C., Courel, J., Delisle, F., Duprat, P., Fouin, L., François, D., Gascon, M.-O., Godineau, D., Grimal, R., Hubert, J.-P., Gal, Y. L., Guennec, J. L., Jeannic, T. L., Longuar, Z., Madre, J.-L., Nicolas, J.-P., Pallez, D., Papon, F., Paulo, C., Quételard, B., Ranty, A., Robin, M., Roux, S., Seguin, S., Siméon, T., de Solère, R., Tisserand, B., et Verry, D. (2010). La mobilité des français, panorama issu de l'enquête nationale transports et déplacements 2008.

[Bareth et al., 2010] Bareth, U., Kupper, A., et Ruppel, P. (2010). geoxmart - a marketplace for geofence-based mobile services. In *Computer Software and Applications Conference (COMPSAC), 2010 IEEE 34th Annual*, pages 101–106.

[Bertaux et al., 2013] Bertaux, A., Cremieux, L., Hausherr, P., Hormaeche, B., Guillet, M., Lerda, M., et Perez, M. (2013). Dossier de post-evaluation.

[Binzenhofer et al., 2006] Binzenhofer, A., Kunzmann, G., et Henjes, R. (2006). A scalable algorithm to monitor chord-based p2p systems at runtime. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 8 pp.–.

[Boissier, 2013] Boissier, O. (2013). Multi-agent oriented programming and intelligent environments. In *Control Systems and Computer Science (CSCS), 2013 19th International Conference on*, pages 457–464.

[Bond et Gasser, 1988] Bond, A. H. et Gasser, L. (1988). *A survey of distributed artificial intelligence.* Readings in Distributed Artificial Intelligence. Morgan Kaufmann Publishers, San Mateo CA.

[Bousselmi et Amara, 2011] Bousselmi, A. et Amara, K. (2011). Rapport de projet de fin d'études. entre Ecole Centrale de Lille et Ecole Nationale d'Ingénieurs de Gabes.

[Bousselmi et al., 2012] Bousselmi, A., Zgaya, H., Bourdeaud'Huy, T., et Hammadi, S. (2012). Architecture à base d'agents pour optimiser les services d'aide à la mobilité : vers une conception d'un espace ubiquitaire. In *MajecSTIC 2012*, Lille, France.

[Bousselmi et al., 2014] Bousselmi, A., Zgaya, H., Bourdeaud'huy, T., et Hammadi, S. (2014). A based dynamic role-enabled multi-agent information system. In *The 2nd AAMAS Workshop on Collaborative Online Organization.*

[Bousselmi et al., 2013] Bousselmi, A., Zgaya, H., Bourdeaud' huy, T., et Hammadi, S. (2013). Innovative service-oriented information system for supporting mobility in ubiquitous spaces : A role driven

approach. *International Journal of Management and Information Technology. IJMIT*, 5(2) :488–502.

[Burin des Roziers et al., 2011] Burin des Roziers, C., Chelius, G., Ducrocq, T., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N., Noél, T., et Vandaele, J. (2011). Using senslab as a first class scientific tool for large scale wireless sensor network experiments. In *NETWORKING 2011*, pages 147–159. Springer Berlin Heidelberg.

[Cai et al., 2013] Cai, Z., Chen, C., et Wang, Q. (2013). Architecture design of mobile cloud and prototype test. In *Communications and Networking in China (CHINACOM), 2013 8th International ICST Conference on*, pages 615–621.

[Camarasu-Pop et al., 2013] Camarasu-Pop, S., Glatard, T., et Benoit-Cattin, H. (2013). Simulating application workflows and services deployed on the european grid infrastructure. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pages 18–25.

[Cappello et al., 2005] Cappello, F., Caron, E., Dayde, M., Desprez, F., Jegou, Y., Primet, P., Jeannot, E., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Quetier, B., et Richard, O. (2005). Grid'5000 : a large scale and highly reconfigurable grid experimental testbed. In *Grid Computing, 2005. The 6th IEEE/ACM International Workshop on*, pages 8 pp.–.

[Cardon et al., 2000] Cardon, A., Galinho, T., et Vacher, J.-P. (2000). Genetic algorithms using multiobjectives in a multi-agent system. *Robotics and Autonomous Systems*, 33(2–3) :179 – 190.

[Chakroun, 2013] Chakroun, I. (2013). *Algorithmes Branch and Bound parallèles hétérogènes pour environnements multi-coeurs et multi-GPU*. PhD thesis, Université des Sciences et Technologie de Lille.

[Collette et Siarry, 2002] Collette, Y. et Siarry, P. (2002). *Optimisation multiobjectif*. Algorithmes. Eyrolles, Paris.

[Cossentino et al., 2009] Cossentino, M., Gaud, N., Hilaire, V., Galland, S., et Koukam, A. (2009). Aspecs : an agent-oriented software process for engineering complex systems.

[Dabek, 2005] Dabek, F. (2005). *A Distributed Hash Table*. PhD thesis, Massachusetts Institute of Technology.

[Debes et al., 2005] Debes, M., Lewandowska, A., et Seitz, J. (2005). Definition and implementation of context information. In *Proceedings Of The 2nd Workshop On Positioning, Navigation And Communication (WPNC'05) and 1st Ultra-Wideband Expert Talk (UET'05)*, WPNC'05 and UET'05.

[Desruelle et al., 2010] Desruelle, H., Gielen, F., et Lyle, J. (2010). Testing self-adaptive applications with simulation of context events. In *Proceedings of the 3rd International DisCoTec Workshop on Context-Aware Adaptation Mechanisms for Pervasive and Ubiquitous Services*, CAMPUS '10.

[Durmus et Erdogan, 2009] Durmus, A. et Erdogan, N. (2009). A web services market framework with role based agents. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 4722–4727.

[Farkas et al., 2011] Farkas, G., Szanto, I., Gora, V., et Haller, P. (2011). Extending the boinc architecture using peer-to-peer application code exchange. In *Roedunet International Conference (RoEduNet), 2011 10th*, pages 1–4.

[Fathy et al., 2012] Fathy, M., GholamalitabarFirouzjaee, S., et Raahemifar, K. (2012). Improving qos in {VANET} using {MPLS}. *Procedia Computer Science*, 10(0) :1018 – 1025. {ANT} 2012 and MobiWIS 2012.

[Feki, 2010] Feki, M. F. (2010). *Distributed optimization for multi-operator routes search in co-modal transport network*. PhD thesis, Ecole Centrale de Lille.

[Ferber et Perrot, 1995] Ferber, J. et Perrot, J.-F. (1995). *Les systèmes multi-agents : vers une intelligence collective*. IIA, Informatique, intelligence artificielle. InterEditions, Paris.

[Fernando et al., 2013] Fernando, N., Loke, S. W., et Rahayu, W. (2013). Mobile cloud computing : A survey. *Future Generation Computer Systems*, 29(1) :84 – 106.

[Ferrari et Zhu, 2011] Ferrari, L. et Zhu, H. (2011). Enabling dynamic roles for agents. In *Collaboration Technologies and Systems (CTS), 2011 International Conference on*, pages 500–507.

[Fides-Valero et al., 2008] Fides-Valero, A., Freddi, M., Furfari, F., et Tazari, M.-R. (2008). The persona framework for supporting context-awareness in open distributed systems. In *Ambient Intelligence*, volume 5355 of *Lecture Notes in Computer Science*, pages 91–108.

[Foster et al., 2001] Foster, I., Kesselman, C., et Tuecke, S. (2001). The anatomy of the grid : Enabling scalable virtual organizations. *Int. J. High Perform. Comput. Appl.*, 15(3) :200–222.

[Friedewald et Raabe, 2011] Friedewald, M. et Raabe, O. (2011). Ubiquitous computing : An overview of technology impacts. *Telematics and Informatics*, 28(2) :55 – 65.

[Gaiti, 1993] Gaiti, D. (1993). Intelligent distributed systems : New trends. In *Distributed Computing Systems, 1993., Proceedings of the Fourth Workshop on Future Trends of*, pages 106–111.

[Giorgino et al., 2010] Giorgino, T., Harvey, M., et de Fabritiis, G. (2010). Distributed computing as a virtual supercomputer : Tools to run and manage large-scale {BOINC} simulations. *Computer Physics Communications*, 181(8) :1402 – 1409.

[Groß et al., 2013] Groß, C., Richerzhagen, B., Stingl, D., Weber, J., Hausheer, D., et Steinmetz, R. (2013). Geoswarm : A multi-source download scheme for peer-to-peer location-based services. In IEEE, editor, *Proceedings of the 13th IEEE Conference on Peer-to-Peer Computing*.

[Han et al., 2014] Han, J., Chung, T., Kim, S., chul Kim, H., Kangasharju, J., Kwon, T. T., et Choi, Y. (2014). Strategic bundling for content availability and fast distribution in bittorrent. *Computer Communications*, 43 :64 – 73.

[Harrabi et al., 2013] Harrabi, S., Chainbi, W., et Ghedira, K. (2013). A multi-agent approach for routing on vehicular ad-hoc networks. *Procedia Computer Science*, 19(0) :578 – 585. The 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd International Conference on Sustainable Energy Information Technology (SEIT-2013).

[Hayes, 2008] Hayes, B. (2008). Cloud computing. *Commun. ACM*, 51(7) :9–11.

[Heo et al., 2010] Heo, G., Kim, E., et Choi, J. (2010). An extended snmp-based management of digital convergence devices. In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 2540–2547.

[Huang et al., 2008] Huang, Y., Fu, T. Z., Chiu, D.-M., Lui, J. C., et Huang, C. (2008). Challenges, design and analysis of a large-scale p2p-vod system. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, SIGCOMM '08, pages 375–388.

[Huhns, 2012] Huhns, M. (2012). *Distributed Artificial Intelligence*. Number vol. 1. Elsevier Science.

[Jennings et Wooldridge, 1996] Jennings, N. et Wooldridge, M. (1996). Software agents. In *IEE Review*, volume 42, pages 17–20.

[Kaddoussi et al., 2009] Kaddoussi, A., Zgaya, H., Hammadi, S., et Bretaudeau, F. (2009). PAAN : Partial Agreement Negotiation Network based on Intelligent Agents in Crisis Situation. *International Journal of Mathematics and Computers in Simulation*, 3(4) :170–178.

[Kang et Sim, 2011] Kang, J. et Sim, K. M. (2011). Towards agents and ontology for cloud service discovery. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on*, pages 483–490.

[Kawahara et al., 2007] Kawahara, Y., Kawanishi, N., Ozawa, M., Morikawa, H., et Asami, T. (2007). Designing a framework for scalable coordination of wireless sensor networks, context information and web services. In *Distributed Computing Systems Workshops, 2007. ICDCSW '07. 27th International Conference on*, pages 44–44.

[Khan et al., 2013] Khan, A. N., Kiah, M. M., Khan, S. U., et Madani, S. A. (2013). Towards secure mobile cloud computing : A survey. *Future Generation Computer Systems*, 29(5) :1278 – 1299.

[Khan, 2010] Khan, M. U. (2010). *Unanticipated Dynamic Adaptation of Mobile Applications*. PhD thesis, Kassel University.

[Kotwal et Singh, 2012] Kotwal, P. et Singh, A. (2012). Evolution and effects of mobile cloud computing, middleware services on cloud, future prospects : A peek into the mobile cloud operating systems. In *Computational Intelligence Computing Research (ICCIC), 2012 IEEE International Conference on*, pages 1–5.

[Ku et al., 2014] Ku, I., Lu, Y., et Gerla, M. (2014). Software-defined mobile cloud : Architecture, services and use cases. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International*, pages 1–6.

[Laforenza, 2006] Laforenza, D. (2006). European strategies towards next generation grids. In *Parallel and Distributed Computing, 2006. ISPDC '06. The Fifth International Symposium on*, pages 11–11.

[Land, 2004] Land, R. (2004). Understanding evolution of information systems by applying the general definition of information. In *Information Technology Interfaces, 2004. 26th International Conference on*, pages 447–452 Vol.1.

[Lind, 2001] Lind, J. (2001). Issues in agent-oriented software engineering. In Ciancarini, P. et Wooldridge, M. J., editors, *Agent-Oriented Software Engineering*, volume 1957 of *Lecture Notes in Computer Science*, pages 45–58. Springer Berlin Heidelberg.

[Liu et al., 2009] Liu, F., Li, L., et Chou, W. (2009). Communications enablement of software-as-a-service (saas) applications. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–8.

[Liu et al., 2005] Liu, J., Gu, N., Zong, Y., Ding, Z., Zhang, S., et Zhang, Q. (2005). Service registration and discovery in a domain-oriented uddi registry. In *Computer and Information Technology, 2005. CIT 2005. The Fifth International Conference on*, pages 276–283.

[Liu et al., 2014] Liu, T., Ceder, A. A., Ma, J., et Guan, W. (2014). Cbvc-b : A system for synchronizing public-transport transfers using vehicle-to-vehicle communication. *Procedia - Social and Behavioral Sciences*, 138(0) :241 – 250. The 9th International Conference on Traffic and Transportation Studies (ICTTS 2014).

[Lua et al., 2005] Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., et Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, 7 :72–93.

[Margery et al., 2014] Margery, D., Morel, E., Nussbaum, L., Richard, O., et Rohr, C. (2014). Resources Description, Selection, Reservation and Verification on a Large-scale Testbed. In *TRIDENTCOM - 9th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*.

[Marinescu, 2013] Marinescu, D. C. (2013). Cloud computing, theory and practice. In *Cloud Computing, Theory and Practice*. Morgan Kaufmann, Boston.

[Martin et al., 2014] Martin, S., Buchert, T., Willemet, P., Richard, O., Jeanvoine, E., et Nussbaum, L. (2014). Scalable and Reliable Data Broadcast with Kascade. In *HPDIC - International Workshop on High Performance Data Intensive Computing, in conjunction with IEEE IPDPS 2014*.

[Mathieu et Verrons, 2003] Mathieu, P. et Verrons, M.-H. (2003). A generic model for contract negotiation. In *Proceedings of the Artificial Intelligence and the Simulation of Behaviour symposium on Intelligent Agents in Virtual Markets (AISB'02)*, pages 1–8, undef, France.

[Maymounkov et Mazières, 2002] Maymounkov, P. et Mazières, D. (2002). Kademlia : A peer-to-peer information system based on the xor metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 53–65.

[Meng et al., 2007] Meng, A., Ye, L., Roy, D., et Padilla, P. (2007). Genetic algorithm based multiagent system applied to test generation. *Computers and Education*, 49(4) :1205 – 1223.

[Merabti et El Rhalibi, 2004] Merabti, M. et El Rhalibi, A. (2004). Peer-to-peer architecture and protocol for a massively multiplayer online game. In *Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004. IEEE*, pages 519–528.

[Montoya et al., 2010] Montoya, A., Restrepo, D., et Ovalle, D. (2010). Artificial intelligence for wireless sensor networks enhancement. In *Smart Wireless Sensor Networks, Yen Kheng Tan (Ed.)*, Lille, France.

[Nguyen et al., 2013] Nguyen, K.-K., Cheriet, M., et Lemay, M. (2013). Enabling infrastructure as a service (iaas) on ip networks : from distributed to virtualized control plane. *Communications Magazine, IEEE*, 51(1) :136–144.

[Nikolai et Madey, 2009] Nikolai, C. et Madey, G. (2009). Tools of the trade : A survey of various agent based modeling platforms. *Journal of Artificial Societies and Social Simulation*, 12(2) :2.

[Nilsson, 1981] Nilsson, N. J. (1981). Distributed artificial intelligence. In *SRI International Menlo Park CA Artificial Intelligence Center*.

[Ohashi, 2010] Ohashi, M. (2010). Japanese ubiquotous network project : Ubila. In Nakashima, H., Aghajan, H., et Augusto, J., editors, *Handbook of Ambient Intelligence and Smart Environments*, pages 1229–1255. Springer US.

[Paik et al., 2014] Paik, I., Chen, W., et Huhns, M. (2014). A scalable architecture for automatic service composition. *Services Computing, IEEE Transactions on*, 7(1) :82–95.

[Paliwal et al., 2012] Paliwal, A., Shafiq, B., Vaidya, J., Xiong, H., et Adam, N. (2012). Semantics-based automated service discovery. *Services Computing, IEEE Transactions on OC*, 5(2) :260–275.

[Pesonen et Horster, 2012] Pesonen, J. et Horster, E. (2012). Near field communication technology in tourism. *Tourism Management Perspectives*, 4(0) :11 – 18.

[Portmann et al., 2001] Portmann, M., Sookavatana, P., Ardon, S., et Seneviratne, A. (2001). The cost of peer discovery and searching in the gnutella peer-to-peer file sharing protocol. In *Networks, 2001. Proceedings. Ninth IEEE International Conference on*, pages 263–268.

[Pourebrahimi et al., 2005] Pourebrahimi, B., Bertels, K., et Vassiliadis, S. (2005). A survey of peer-to-peer networks. In *Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Proessing*.

[Pouwelse et al., 2005] Pouwelse, J., Garbacki, P., Epema, D., et Sips, H. (2005). The bittorrent p2p file-sharing system : Measurements and analysis. In *Peer-to-Peer Systems IV*, volume 3640, pages 205–216. Springer Berlin Heidelberg.

[Pujalte-Busseuil et al., 2006] Pujalte-Busseuil, V., Ramadour, P., et Tranvouez, E. (2006). Conception par réutilisation : des composants aux agents composants. In *Actes du XXIVème Congrès INFORSID, Hammamet, Tunisie, 31 mai - 4 juin, 2006*, pages 1–16.

[Rajakaruna et al., 2012] Rajakaruna, G., Karunananda, A., et Jayalal, S. (2012). Aligning ontologies using multi agent technology. In *Advances in ICT for Emerging Regions (ICTer), 2012 International Conference on*, pages 189–196.

[Roussopoulos et al., 2004] Roussopoulos, M., Baker, M., Rosenthal, D., Guili, T., Maniatis, P., et Mogul, J. (2004). 2 p2p of not 2 p2p ? In *International workshop on Peer-To-Peer Systems, IPTPS 2004*.

[Rouvoy et al., 2008] Rouvoy, R., Beauvois, M., Lozano, L., Lorenzo, J., et Eliassen, F. (2008). Music : An autonomous platform supporting self-adaptive mobile applications. In *Proceedings of the 1st Workshop on Mobile Middleware : Embracing the Personal Communication Device*, MobMid '08, pages 6 :1–6 :6.

[Ruta et al., 2014] Ruta, M., Scioscia, F., Loseto, G., et Di Sciascio, E. (2014). Semantic-based resource discovery and orchestration in home and building automation : A multi-agent approach. *Industrial Informatics, IEEE Transactions on*, 10(1) :730–741.

[Saad et al., 2008] Saad, S., Zgaya, H., et Hammadi, S. (2008). The flexible negotiation ontology-based knowledge management system : The transport ontology case study. In *Information and*

*Communication Technologies : From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*, pages 1–7.

[Salma, 2014] Salma, N. (2014). *Adaptation des services sensibles au contexte selon une approche intentionnelle*. PhD thesis, Université Panthéon-Sorbonne - Paris I.

[Satunin et Babkin, 2014] Satunin, S. et Babkin, E. (2014). A multi-agent approach to intelligent transportation systems modeling with combinatorial auctions. *Expert Systems with Applications*, 41(15) :6622 – 6633.

[Satyanarayanan et al., 2009] Satyanarayanan, M., Bahl, P., Caceres, R., et Davies, N. (2009). The case for vm-based cloudlets in mobile computing. *Pervasive Computing, IEEE*, 8(4) :14–23.

[Schollmeier, 2001] Schollmeier, R. (2001). A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Peer-to-Peer Computing, 2001. Proceedings. First International Conference on , vol., no.,*.

[Sghaier, 2011] Sghaier, M. (2011). *A combination of optimization and distributed artificial intelligence techniques to set up a dynamic carpooling service*. PhD thesis, Ecole Centrale de Lille.

[Shoham, 1997] Shoham, Y. (1997). Software agents. chapter An Overview of Agent-oriented Programming, pages 271–290. MIT Press.

[Simanta et al., 2012] Simanta, S., Lewis, G., Morris, E., Ha, K., et Satyanarayanan, M. (2012). A reference architecture for mobile code offload in hostile environments. In *Software Architecture (WICSA) and European Conference on Software Architecture (ECSA), 2012 Joint Working IEEE/IFIP Conference on*, pages 282–286.

[Smith, 1980] Smith, R. G. (1980). The contract net protocol : High-level communication and control in a distributed problem solver. *Computers, IEEE Transactions on*, C-29(12) :1104–1113.

[Sordoni et al., 2009] Sordoni, A., Briot, J.-P., Sebba Patto, V., Vasconcelos, E., et Irving, M. (2009). Automatisation par agentification d'un gestionnaire de parc dans le cadre du projet SimParc. In *Colloque Modèles et Apprentissage en Sciences Humaines et Sociales (MASH'09)*. Institut de Mathématiques de Toulouse & Institut de Recherche en Informatique de Toulouse.

[Stajano, 2010] Stajano, F. (2010). Security issues in ubiquitous computing*. In Nakashima, H., Aghajan, H., et Augusto, J., editors, *Handbook of Ambient Intelligence and Smart Environments*, pages 281–314. Springer US.

[Stoica et al., 2001] Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., et Balakrishnan, H. (2001). Chord : A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM'01*, pages 149–160.

[Sun, 2001] Sun, J.-Z. (2001). Mobile ad hoc networking : an essential technology for pervasive computing. In *Info-tech and Info-net, 2001. Proceedings. ICII 2001 - Beijing. 2001 International Conferences on*, volume 3, pages 316–321 vol.3.

[Tamilarasi et Ramakrishnan, 2012] Tamilarasi, K. et Ramakrishnan, M. (2012). Design of an intelligent search engine-based uddi for web service discovery. In *Recent Trends In Information Technology (ICRTIT), 2012 International Conference on*, pages 520–525.

[Tazari, 2010] Tazari, M.-R. (2010). An open distributed framework for adaptive user interaction in ambient intelligence. In *Ambient Intelligence*, volume 6439, pages 227–238. Springer Berlin Heidelberg.

[Tsukada, 2011] Tsukada, M. (2011). *Communications Management in Cooperative Intelligent Transportation Systems*. PhD thesis, MINES ParisTech.

[Uzun et al., 2013] Uzun, A., Salem, M., et Kupper, A. (2013). Semantic positioning - an innovative approach for providing location-based services based on the web of data. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 268–273.

[Valencio et al., 2013] Valencio, C., Ferreira Da Silva, A., Lemos Guimaraes, D., Mauro Cansian, A., Tronco, M., et Roschildt Pinto, A. (2013). Gbd iaas manager : A tool for managing infrastructure-as-a-service for private and hybrid clouds. In *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2013 International Conference on*, pages 206–211.

[Weiser, 1993] Weiser, M. (1993). Hot topics-ubiquitous computing. *Computer*, 26(10) :71–72.

[Wooldridge, 2009] Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition.

[Wooldridge et al., 2000] Wooldridge, M., Jennings, N. R., et Kinny, D. (2000). The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, 3(3) :285–312.

[Woolridge, 2001] Woolridge, M. (2001). *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA.

[Yang et Garcia-Molina, 2001] Yang, B. et Garcia-Molina, H. (2001). Comparing hybrid peer-to-peer systems. In *Proceedings of the 27th International Conference on Very Large Data Bases*, VLDB '01, pages 561–570.

[Yassine, 2011] Yassine, E. G. (2011). *La Sensibilité au contexte dans un environnement mobile*. PhD thesis, Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes (ENSIAS), Rabat.

[Zgaya, 2005] Zgaya, H. (2005). Workplan mobile agent for the transport network application. In *IMACS World Congress Scientific Computation Applied Mathematics and Simulation*.

[Zgaya, 2007] Zgaya, H. (2007). *Design and distributed optimization of an information system for the urban mobility assistance : A multi-agent approach for the search and the composition of the services related to the transport.* PhD thesis, Ecole Centrale de Lille.

[Zgaya et Hammadi, 2006] Zgaya, H. et Hammadi, S. (2006). Assignment and integration of distributed transport services in agent-based architecture. In *Intelligent Agent Technology, 2006. IAT06. IEEE/WIC/ACM International Conference on.*

[Zgaya et Hammadi, 2007] Zgaya, H. et Hammadi, S. (2007). Multi-agent information system using mobile agent negotiation based on a flexible transport ontology. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07. ACM.

[Zgaya et al., 2008] Zgaya, H., Tang, D., Hammadi, S., et Bretaudeau, F. (2008). Negotiation protocol according to the perturbation impact in a multi-agent supply chain system for the crisis management. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, volume 3, pages 698–701.

[Zhang et al., 2009] Zhang, B., Iosup, A., Pouwelse, J., Epema, D., et Sips, H. (2009). On assessing measurement accuracy in bittorrent peer-to-peer file-sharing networks.

[Zhang et al., 2010] Zhang, B., Iosup, A., Pouwelse, J., Epema, D., et Sips, H. (2010). Sampling bias in bittorrent measurements. In *European Conference on Parallel Processing (Euro-Par'10)*, volume 6271, pages 484–496. Springer.

[Zhu et al., 2010] Zhu, H., Hou, M., et Zhou, M. (2010). Establishing the foundation of adaptive collaboration. In *Collaborative Technologies and Systems (CTS), 2010 International Symposium on*, pages 546–554.

[Zhu et Zhou, 2008] Zhu, H. et Zhou, M. (2008). Roles in information systems : A survey. *Systems, Man, and Cybernetics, Part C : Applications and Reviews, IEEE Transactions on*, 38(3) :377–396.

[Zhu et Zhou, 2010] Zhu, H. et Zhou, M. (2010). Issues in adaptive collaboration. In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 2828–2833.

[Zidi, 2006] Zidi, K. (2006). *Interactive system of Help in Multimodal travels.* PhD thesis, Ecole Centrale de Lille ; Université des Sciences et Technologie de Lille - Lille I.

# Index

# Special Thanks

Un très grand merci à : Mohamed Moez Belhaj, Zhanjun Wang, Wided Chandoul, Safa Gharbi, Astrid Piconese, Hassan Omran.

Je tiens à remercier également : imen h., moussa c., mohamed salah k., khalil a., ismail y., kahina a., oussama b., maher f., yassine s., ahmed s., sodes b. c., sarah o., ines a., lotfi z., mayank j., souheil o., noex b., nora w., nouha j., sonia b., abdou h., fekri e., achref y., mohamed c., wissem c., wiem b., imen n., ahmed j., rayen w., haythem d., aymen k., koutaiba r., abdelrahim m., majdi b., adnen e. a., rahma l., issam k., sofien b., ramzi b. m., farouk t., mohamed b., fadoua b. t., raoul d., bilel b., marwa b., ahmed m.

***Conception et développement d'un système d'information d'aide à la mobilité : une approche multi-agent pour la recherche et la composition des services dans un espace ubiquitaire***

***Résumé :*** Dans un contexte de mobilité ubiquitaire, des différents objets sont capables d'interagir avec les utilisateurs pour leurs fournir des services innovants et les aider à optimiser leurs plans de déplacement. En effet, le nombre des utilisateurs est ainsi que le nombre de fournisseurs de service demandés par ces utilisateurs sont en pleine augmentation. Cette croissance implique un aspect de concurrence et nécessite des choix optimisés. Dans ce cadre, l'objectif de cette thèse est de concevoir et optimiser un système d'aide au déplacement qui couvre non seulement les services de déplacement quotidien mais aussi les services touristiques, culturels et bien d'autres. Les travaux de recherche présentés dans ce manuscrit proposent la mise en place d'une Plateforme de Recherche et de composition des Services d'Aide à la Mobilité (PRoSAM) afin d'optimiser les tâches de recherche, composition et distribution des Informations de Mobilité Avancée (IMA). L'aspect dynamique et distribué du problème, nous a conduit à adopter une modélisation orientée agent afin de s'adapter aux conditions d'un environnement ubiquitaire. Grâce à une approche de changement dynamique des rôles des agents utilisateurs et un protocole de négociation innovant, les clients sont capables d'échanger les services d'une manière autonome et d'établir des accords totaux ou partiels en fonction de l'offre et de la demande. Finalement, les résultats de simulation présentés dans cette thèse démontrent l'efficacité des approches adoptées.

***Mots Clés :*** Système d'information, aide à la mobilité avancée, système multi-agent, optimisation génétique, informatique ubiquitaire

***Design and development of a mobility aid information system : a agent-based approach for searching and composing services in a ubiquitous space***

***Abstract :*** In a context of ubiquitous mobility, different objects are able to interact with users to provide them with innovative services and help them optimize their travel plans. Indeed, the number of users and the number of service providers requested by these users are actively growing. This growth involves an aspect of competition and requires optimized choices. In this context, the aim of this thesis is to design and optimize a mobility aid system that covers not only transport services but also tourist services, cultural services and many others. The research presented in this thesis proposes the establishment of a Plateforme de Recherche et de composition des Services d'Aide à la Mobilité (PRoSAM) to optimize research, composition and distribution tasks of advanced mobility information. The dynamic and distributed aspects of the problem have led us to adopt an agent-oriented modeling approach to cope with the conditions of a ubiquitous environment. Thanks to a dynamic role switching strategy of user agents and an innovative negotiation protocol, customers are able to exchange services autonomously and to establish full or partial agreements in order to optimize communications over the network. Finally, the simulation results presented in this thesis demonstrate the effectiveness of the proposed approaches.

***Keywords :*** Information system, Advanced mobility aid, Multi-agent system, Genetic optimization, Ubiquitous computing