ISSN 2278-5612

Innovative Service-Oriented Information System for supporting Mobility in Ubiquitous Spaces: A Role Driven Approach

Ayoub BOUSSELMI, Hayfa ZGAYA, Thomas BOUREDEAUD'HUY, Slim HAMMADI
Ecole Centrale de Lille, Cité Scientifique – BP 48, 59651 Villeneuve D'Ascq cedex, France
ayoub.bousselmi@ec-lille.fr
Université Lille 2 Droit et Santé, Institut Lillois d'Ingénierie de la Santé, 42 rue Ambroise Paré, 59120 LOOS - France
hayfa.zgaya@univ-lille2.fr
Ecole Centrale de Lille, Cité Scientifique – BP 48, 59651 Villeneuve D'Ascq cedex, France
thomas.bourdeaud_huy@ec-lille.fr
Ecole Centrale de Lille, Cité Scientifique – BP 48, 59651 Villeneuve D'Ascq cedex, France
slim.hammadi@ec-lille.fr

ABSTRACT

Conventional architectures based on a central servers are not suitable and cannot scale well in a distributed and high dynamic computing enviroment. Futuristic ubiquitous landscape requires the development of robust, reliable and flexible distributed information systems capable of large scaling and supporting the large amount of simultaneous user requests. In this paper, we propose an innovative information system based on intelligent agents, which is capable of optimizing the search task and managing the services within a coalition of agents. Users will request a set of information services provided by information providers distributed among a Big Data Network (BDN). The system is divided into two subsystems. The first subsystem (Connector) is responsible for the search optimization task using an evolutionary algorithm. It generates optimized Workpans for mobile agents that collect needed data from the BDN. The scond subsystem (Coalition) is composed of of peer agents whitch are endowed with the ability to switch dynamically their role. We also present an effective comparison between two types of architectures: the SRBA (static role-based architecture) and the DBRA (dynamic role-based architecture). We provide in the end of this paper simulations supporting our claims about role switching performances and statistical results about the coalition of agents before and after role switching.

Keywords

Multi-agent system, ubiquitous computing, distributed optimization, big data networks, peer-to-peer system, role switching, mobility, information system.

Council for Innovative Research
Peer Review Research Publishing System
Journal: International Journal of Management & Information Technology
Vol. 5, No. 2
editor@cirworld.com
www.cirworld.com, member.cirworld.com

ISSN 2278-5612

1. INTRODUCTION

In conventional information systems, typically a central server or cluster of servers is essential to fulfill the following tasks: clients authentication, data storage and management, web-services and applications supply, routing services for other providers, searching and discovering services for ordinary clients or business-to-business clients, publishing services for service providers, etc. Yet, not only the number of users that grows exponentially but also the data amount and information flow. Furthermore, additional challenges are presented by information and communication technologies (ICT) in a ubiquitous space. In fact, the use of mobile devices is increasing in our daily activities. A communicant mobile device (e.g. Smartphone, tablet, laptop, PDA, etc.) can be considered as an appropriate implementation of the ubiquitous terminal (UT) of the ubiquitous computing paradigm. The UT is allowing people to be connected to different information systems (e.g. Metropolitan Area Network, Wide Area Network, Internet, WLAN, etc.) anywhere and anytime. Given this context of permanent and flexible access to a massive data warehouse, getting relevant information becomes crucial. We think that a mediator system in charge of optimizing search queries and maintaining system's load in an efficient level is unavoidable.

UTs are devices with limited resources and use generally wireless communication technologies (e.g. Wi-Fi IEEE 802.11, ZigBee IEEE 802.15.4, etc.). This makes the development of reliable and robust information systems a considerable challenge. Therefore, parameters that may interpret some particular problematic characteristics such as autonomy, memory and so on, should be taken into account when modeling a ubiquitous information system. Other challenges are presented in such system principally the large scaling and the service discovery processes. In fact, to build a completely distributed system capable of large scaling, nodes (UTs), at some point, should not require any particular node to access system services. For security reason, we think that it is better to centralize some tasks such as authentication. The service discovery may be implemented using different technologies such as yello pages or hash tables.

Besides, new distributed applications (data minig, information extraction, data integration, etc.) are spread on huge networks, employing heterogeneous and voluminous data sources located on geographically and semantically separated sites. Through such networks, different actions (requests, search, storage, update, etc.) could be invoked in a high frequency and simultaneously, requiring the access to several heterogeneous and distant data sources. This behaviour can be simulated and analysed using different kind of interaction among individual computing entities. Thus, a multi-agent approach seems to be the most appropriate to implement this complex character.

Moreover, distributed applications through wide networks are not easy to realize because of the limited aspect of bandwidth which remains restricted by several technical factors. Therefore, permanent connexion can't be constantly available so we have to manage data information availability despite recurrent connexions in order to access and share distributed data located through a Big Data Network (BDN).

In this paper, we propose the PRECSAM (Platform for search and composition of Mobility Enhancement Services), an information system that supports users' mobility in ubiquitous spaces and allows them to access several services in an efficient way. The system architecture is based on agent technology. The system's environment is dynamic because of clients that join and leave dynamically. That's why we need efficient techniques to manage the system. We think that advances in peer-to-peer (P2P) systems are helpful to our problem. We believe that the fundamental question that P2P systems came in response to is how could we outstrip the existence of a central server that receives client requests and responses or redirect them to another specific server capable of providing the needed service.

P2P architectures are capable of self organization in spite of the continuous variation of node populations and the dynamic aspect of the network (nodes connect and leave the network randomly). P2P architectures are capable of large scaling and distribution of the growing access to resources. Opertions such as administration and maintenance no longer belong to a central entity but distributed among system users. Removing the central gate between clients and service providers in P2P fashion provides many advantages in term of:

- scalability: the system will be able to scale and to support a large number of clients, providers and services;
- response time: direct communication between client and service provider becomes faster than communication by means of proxies;
- autonomy: entities that compose the system will be responsible for the maintenance of their own resources;
- accessibility: information will be more accessible because it is no more limited to central entity performance.

The PRECSAM is divided into two subsystems:

- The first subsystem is responsible for the search optimization task. First, simultaneously incoming requests are decomposed into a list of elementary services. Then, similarities between requests are spotted to reduce redundant searches. Requested services could be supplied by different service providers with different costs and quality of service. At this level, an evolutionary algorithm is applied to find an optimized solution (affectation services-providers). After that, mobiles agents are used to collect needed information from the network. Once responses are collected, they will be distributed among user agents.
- The second subsystem is composed of user agents that are considered as a large computation and storage resource and could be used to enhance system's reliability and flexibility. User agents in this case have two roles: service provider role and seeker role.

We provide also a comparison between two types of architectures:

- The static role-based architecture (SRBA);
- The dynamic role-based architecture (DRBA).

We prove through this paper the importance af the role switching strategy. We provide results of many simulation scenarios that prove the correctness of our proposition. The role switching inhances significantly the hole system load causing a high decrease of the response time in a distributed information system.

The paper is structured as follow: a state of the art about ubiquitous computing, multi-agents systems and peer-to-peer networks is presented in the rest of the introduction. Section II details the problem formulation. Section III presents the system overview. Section IV presents simulations supporting our claims about role switching performance. Finally, we summarize our contributions in Section V.

1.1 Ubiquitous computing

Ubiquitous computing (Ubicomp) is an emergent concept that represents a framework for new intelligent human-environment interactions. This is realized throw the massive integration of computers in our daily environment objects enabling the latters to communicate and to be aware of their context. From user point of view, this concept will guarantee “anytime and anywhere” intelligent interactions and provide performant services and applications. According to [21], Ubicomp and virtual reality are two opposite paradigms. In fact, the virtual reality takes persons within a virtual world created by the computer. Although, Ubicomp brings computers to our real word and enrich it with information, services and resources. Ubicomp is also known as “pervasive computing”, “invisible computing”, “disappearing computing”, “sentient computing”, “ambient intelligence” or “calm computing” [19].

In the last decade, many search works and industrial products around the world have been established in concordance with this concept. In 2003, the Japanese government has lunched three national search and development projects: a microchips network project, an authentication network project and a control and administration network project (Ubila) which we will detail it in this survey. Other projects were initiated after that like RFID project in 2004 and Sensor Network project in 2005. The goal of the Ubila project was the foundation of needed technologies for ubiquitous networking and the provision of applications and services support for users. A number of ICT (Information and Communication Technologies) were developed under the Ubila project and we can distinguish two fields [15]. The first is about hardware platforms such as the prototype of inside localization developed within the University of Tokyo named DOLPHIN. This platform is based on distributed ultrasonic detection. Also, a sensor network named PAVENET was developed to support real-time applications to measure seismic vibrations’ acceleration. The second field is about software platforms such as the information system Lifelog developed by the Japanese telecommunications operator KDDI Corporation[1]. This system is based on a centralized data-base in order to provide information about prices, vendors and other characteristics of industrial products. Mobile technologies like GPS, camera, RFID (Radio Frequency IDentification) reader, QR-Code (Query Request Code) reader, etc., are used to communicate with environment’s objects. For efficient communication with low computing capacity devices, the protocol OSNAP was developed on the base of the protocol SNMP (Simple Network Management Protocol). Under the same project Ubila, a system for supporting aware services named CASTANET was developed on the base of the REST (Representational State Transfer) model. It allows gathering contextual data from sensor networks and controlling actuators using HTTP (Hyper Text Transfer Protocol) protocol. CASTANET considers that sensors’ information, web service resources, actuators and users interfaces are all resources that can be accessed through URI (Uniform Resource Identifier). 4 performatives are allowed:

- GET: used for retrieving information about the state of a resource pointed by a given URI
- PUT: used for updating these information
- POST: used for generating resources and sending them to the specified URI
- DELETE: deleting resource saved under the given URI from the server.

The French research project SensLAB [3] [18] is a large scale experimentation platform for wireless sensor networks (WSN). This testbed is distributed among 4 sites, each with the capacity of 256 wireless sensors. This platform provides an application programming interface (API) based also on the REST model. This API helps developers to develop scripts for submitting, reloading and stopping experiments. The supervision of an application is totally independent from the user’s deployed program. In fact, external supervision offers a precise and real-time access to the fundamental parameters of the WSN such as energy consumption and radio activities.

The ubiquitous environment enables real life objects to “think” and to communicate. We believe that the most appropriate unites capable of modeling this advanced artificial intelligence in a highly distributed environment are software agents. The paradigm of computing agency is presented in the next sections.

1.2. Multi-agents Systems

Software Multi-agents systems (MAS) are systems composed of several computer entities in a continueous interaction. These entites are known as agents. Every agent is using local resources and communicates with distant agents. In this section we provide some definitions of MAS and other characteristics.

Agent definition. According to [23], a software agent is a computer system situated in some environment and capable of autonomous actions in order to achieve its delegated goals. According to [6], we call software agent an entity that:

- is situated in an open computer system composed of a set of applications, networks and hetergeneous systems;

[1] http://www.kddi.com/english/

- can communicate with other agents;
- is guided by a set of individual goals or satisfaction function that it tries to optimize;
- have its own resources;
- have a partial representation of other agents;
- have skills (services) that may be requested by other agents;
- in terms of its perception, representations and received communications, it aims to satisfy its goals taking into account its resources and skills.

Agent characteristics. According to [14], an intelligent agent has four main characteristics:

- Autonomy: agents are capable of executing the most part of their task independently and without the intervention of humans or other agents and have a certain degree of control on their own internal states.
- Sociability: an agent is capable of interacting in the right time with other agents or humans to achieve its goals or to give a hand to other agents.
- Reactivity: agents have to sense their environment and respond to changes that might occur.
- Proactivity: agents have to be opportunistic, guided by a goal and take initiative in the right time.

MAS definition. MAS are sets of agents that interact with one another. Generally, agents act on behalf of users and have different goals and motivations. In order to succeed the interaction process, they need to be able to cooperate, coordinate, and negotiate with one another.

MAS are social systems, where agents (individuals) need to interact with each other. Because agents are autonomous entities and make independent decision especially when they have individual goals, their interaction is made generally in an asynchronous fashion. Agents may share a common goal, in this case, tasks among the MAS may be orchestrated (e.g. a global task is divided in a multiple subtasks; each one is assigned to an agent. Subtasks may be independent or there might be some dependencies between them). In the former case, synchronization of the system is needed.

MAS' application areas. This paradigm is useful for our problem because of different reasons:

- MAS are dedicated for geographically distributed problems: this is the case of supply chains control systems or advanced national electronic elections for example.
- MAS are dedicated also for heterogeneous and largely distributed problems: participation of different actors and capacities from different fields. Cooperation in this case is needed in order to achieve a global goal.
- Emergent need for open systems: closed systems are not friendly for interoperability issues. Besides, closed systems can't utilize fully Internet capacities.
- MAS are appropriate for complex problems that can't be resolved with a global vision. MAS offer an advanced resolution strategy based on several local visions that are combined by the mean of communication to get a global vision of the problem.
- The new solutions need to be capable of adaptability and evolution to maintain efficiency and robustness inside a dynamic environment.

MAS' organizational typology. According to [2], agent organizations can be classified by two different relational structures: hierarchical structure and heterarchical structure.

Hierarchical structure: in this type of organizations, control is centralized in a single agent who has the highest level in the hierarchy. This structure is suitable for little organizations and can't scale. It based on a master and a number of slaves that execute its orders.

Heterarchical structure: this type is a kind of spectrum of decentralized organizations that goes from full decentralized systems to less decentralized ones. In fact, we can divide this typology onto three different categories:

- The market organization: this structure is weakly decentralized because of the shared control between several hierarchical coordinator agents which have different goals.
- The community organization: this structure is highly decentralized with no form of hierarchical relations between agents of the system. All agents are with the same weight, the same functionalities and have the same degree of control over the system. This system is free of control nodes and has the capacity to scale.
- The society organization: this type of structure is a mixture of two previous types. Agents may not have the same weights or functionalities; therefore the control degree is different from an agent to another.

Role paradigm in MAS. According to [4], persons are representing specific positions which give them rights ans responsabilities within organizations. Therefor, roles may be defined as a collection of abilities and expected behaviours endowed by who has these positions. A role my be represented by the agent's goal, so agents may aquire and lose roles when their gols change. The role is a convenient paradigm that models organizations, interactions inside groups, communication protocols, hierarchical systems, dependencies between components and resources access rights [24].

Roles are used in many contexts:

- A tool for abstraction, analysis and design of agent-based systems [22] [25]
- Resolution of collaboration and conflict management problems in complex administrative systems [10] [11] [12] [13]
- Assistance of human-machine and machine-machine collaboration
- Modeling of holonic structures for the automation of Workflow process [1]

In the next section, we present the P2P systems and we provide some terminologies. MAS is a convenient tool that to model a high dynamic environment where a large number of interactions are established. Similarly to MAS, P2P systems proven their ability to manage a high amount of data flows and to scale well in a dynamic environment. In the next section we present briefly the definition of a P2P system and we provide details about two applications' specifications.

1.3. Peer-to-peer systems

In the last decade, P2P systems are becoming more and more popular and many industrial products are integrating P2P-like architecture BitTorrent[2], Skype[3], Spotify[4], Viber[5], etc. These systems have proven their ability to support scalability and robustness vis-à-vis the dynamic environment. According to [17] a computer system is considred as a P2P system if it satisfies the three following caracteristics :

- Self-organizing : system nodes organize them selves autonomosly by means of a process of search and discover of the other peers on the network.
- Symmetrical communication : systems nodes are considred as equals and they can be at the same time service seekers or providers in spite of their initial roles (clients or servers).
- Decentralized : system nodes do not need any central entity neither for routing nor management that imposes their individual behaviour. However, peers are autonomous in term of discovering each other, services and available recources on the system.

In literature, many service discovery protocols are proposed. However, few have drawn attention and became references. We present two ewamples of interesting P2P protocols: Chord and Bittorrent

Chord. Chord is based on a well known cryptography technique: hashing. Precisely, it is based on distributed hash tables (DHT). In chord [20], every piece of data stored in the memory of a peer is associated with a key. Each peer is used simultaneously to publish new data on the network using the primitive "*put(key, value)*" and to submit search requests using the primitive "*get(key)*". Peers and data pieces are associated with a hach value called "*id*". Every *id* is composed by m-bits (the *id* of a node can be generated by hashing its IP adress). The topology of the network will have the form of a circle constituted with $N = 2^m$ peers. Every node in the cercle knows its successor (the first node that follows it in the clockwise. To find a key, the request do not have to run through all the network nodes because the system offers another set of rooting information. These information are used to accelerate the process of discovering the key handler. They are stored in a table called "*finger table*". For a key of m-bits, the *finger table* of a node is composed of m lines. The i^{th} line of the table holds the identifier of the first node *s* whitch succeeds n by at least 2^{i-1} on the cercle of identifiers:

$$s = successor\ (n + 2^{i-1})_{1 \le i \le m}$$

The node *s* is called the i^{th} finger of the node *n*. In this way, Chord may guarantee *O(logN)* steps reponses. Figure 1 illustrates the topography of Chord network.

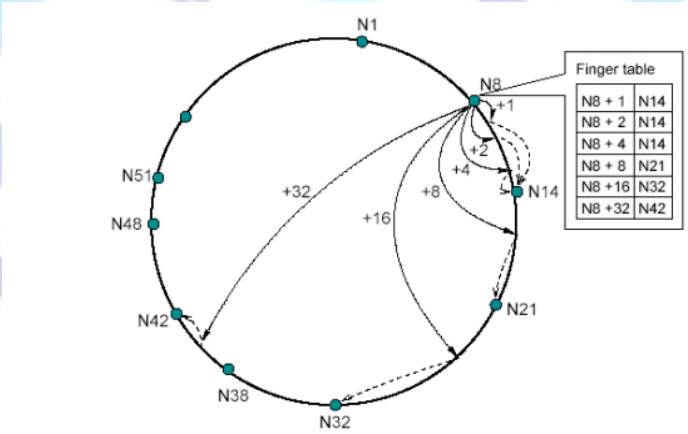

Figure 1. Chord network topography. The successor of the node N_8 by 1, 2 and 4 steps is the node N_{14}, by 8 steps is the node N_{21} and so on.

Bittorrent. Bittorrent is a file download/upload protocol between private users. Exchanged files are divided into thousands of data pieces [16]. Users who are downloading a given file download data pieces that compose the hole file and in the same time upload them to other users in order to eliminate the "selfish" behaviour. Every peer is responsible for maximizing its download ratio by connecting the appropriate peers. Peers which offer a high upload rate will be capable of downloading with high speed. Finally, when a peer finish downloading a file, it could become a "seed" (provider) by

[2] http://www.bittorrent.com/
[3] http://www.skype.com/en/
[4] https://www.spotify.com/fr/
[5] http://www.viber.com/

remaining connected and proposing a free share of the file. In order to find a given file with Bittorrent, users need to access spetial websites which are a kind of global directory of all available files. Then, they need to download a metadata file (*.torrent) which contains all the references related to the needed file and using a Bittorrent client they launch the download process. New content is injected manually on the system through moderators who are responsible for eliminating fake, illegal or very poor quality files.

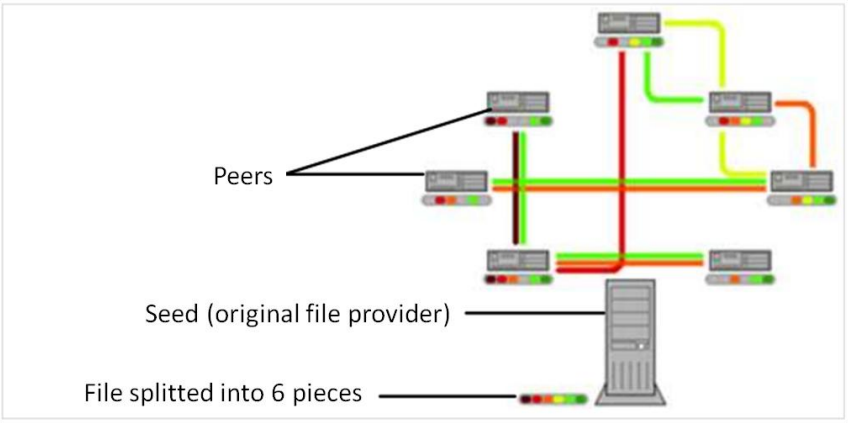

Figure 2. Simplified example of file's pieces destribution using Bittorrent protocol.

In the next section we provide the problem formulation that takes into account system characteristics, particularly, the UT's properties such as autonomy and available memory space.

2. PROBLEM FORMULATION

The aim of this platform is to satisfy its users, optimize the search of services, minimize service's cost and response delay, and maximize service quality. The system is composed of Connectors and Coalitions. Connectors are responsible for yellow page service and optimization tasks. Coalitions are responsible for users' interaction and offers a reliable platform for service sharing in a P2P fashion. Connectors are important because they handle service providers' registration and user access to the system. Coalitions are robustness enhancers because they essentially reduce load on the original service providers and have also a local optimization task related to the ubiquitous terminal status in term of autonomy, memory and computing capabilities. We define the problem as follows:

- a set of **n** agents, each one of them is representing a client in the network at the instant **t**. This set is noted: $\mathbf{A_t = \{a_1, a_2, a_3 \dots a_n\}}$;
- a set of **n** mobile terminals (witch we will call them nodes) responsible for hosting agents and services, and communicating with the rest of nodes in the network at the instant **t**. This set is noted: $\mathbf{T_t = \{T_1, T_2, T_3 \dots T_n\}}$;
- each mobile terminal $\mathbf{T_n}$ is characterized by: an identifier **idTn**, an autonomy indicator $\mathbf{I_A^n}$, a memory space indicator $\mathbf{I_M^n}$, with $\mathbf{I_A^n \in [I_A^{min}, I_A^{max}]}$ et $\mathbf{I_M^n \in [I_M^{min}, I_M^{max}]}$
- a set of **w** connectors responsible for the authentication of clients when they join to the network, for the optimization of the search of services and for the registration of new service providers. This set is noted: $\mathbf{C = \{C_1, C_2, C_3 \dots C_w\}}$;
- a set of **k** servers responsible for offering the requested services. This set is noted: $\mathbf{F = \{F_1, F_2, F_3 \dots F_k\}}$;
- each proposed service $\mathbf{s_j}$ proposed a service provider $\mathbf{F_k}$ is characterized by: an identifier **idSjk**, a quality of service $\mathbf{QoS_j^k}$, a cost $\mathbf{C_j^k}$, a processing delay $\mathbf{P_j^k}$, a quantity of data $\mathbf{Q_j^k}$, and an index of popularity $\mathbf{I_p^j}$. The set of independent elementary services is noted $\mathbf{S = \{s_1, s_2 \dots s_m\}}$;
- a set of **r** simultaneous requests, waiting for responses at the instant **t**. The set of these requests is noted $\mathbf{R_t}$;
- each request $\mathbf{rq_{i \in [1, r]} \in R_t}$ is decomposed into a set of independent elementary services $\mathbf{s_j}$;
- each request $\mathbf{rq_i}$ has a due date $\mathbf{d_i}$ known in advance and a total cost $\mathbf{Co_i}$;

A client query is composed of a set of services. Each service might be provided competitively by a number of servers with different cost, processing time and quality of services (figure 1).

ISSN 2278-5612

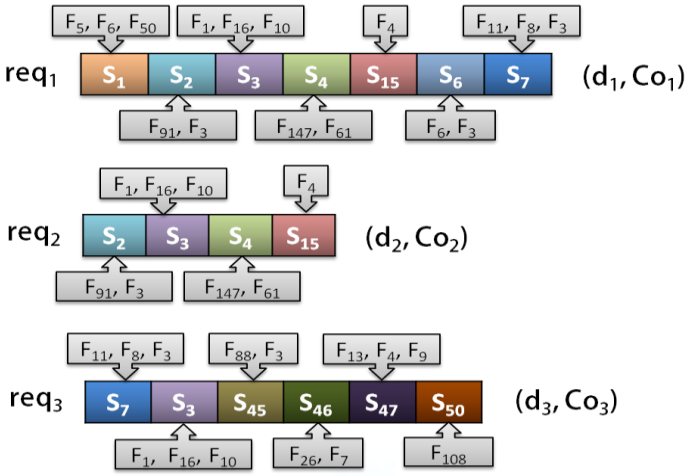

Figure 3. Sample of providers' identification

C_j^k, P_j^k et Q_j^k reprensent the first, second and third terms of each element of the service table below. Note that if a service provider does not provide a given service (partial flexibility), then the correspondent term on the service table is (0, 0, 0). It also possible to have $C_j^k = 0$, $P_j^k \neq 0$ et $Q_j^k \neq 0$ if the information is free.

Table 1. Example of service table.

	F_1	F_2	F_3	…	F_k
s_1	(0 ; 0 ; 0)	(5 ; 0.2 ; 3)	(3 ; 0.6 ; 3)	…	(5 ; 0.3 ; 3)
s_2	(4 ; 0.4 ; 5)	(5 ; 0.1 ; 2)	(5 ; 0.4 ; 1)	…	(8 ; 0.2 ; 3)
s_3	(0 ; 0.5 ; 3)	(0 ; 0 ; 0)	(0 ; 0.6 ; 3)	…	(2 ; 0.8 ; 2)
s_4	(2 ; 0.7 ; 1)	(1 ; 0.4 ; 1)	(0 ; 0 ; 0)	…	(0 ; 0 ; 0)
…	…	…	…	…	…
s_j	(3 ; 1 ; 16)	(1 ; 0.5 ; 3)	(5 ; 0.4 ; 2)	…	(5 ; 0.9 ; 3)

3. SYSTEM OVERVIEW

In this section we present the building locks of our systems. We expose the proposed architecture composed of Connectors and Coalitions of client agents. The system is designed in a way that it will be capable of supporting high number of simultaneous requests. In fact client agents are capable of switching their role to become service providers of a given service once they receive it. The structure of the connectors, and the optimization opproach are also presented.

3.1. System Architecture

The system architecture is designed to offer a robust and flexible service support within futuristic ubiquitous environments. In order to achieve this goal, a set of methods and techniques are combined and introduced in our system. Previous works on information systems for mobility enhancement based on agent technology consider the user as a simple client sitting there and requesting services which raises considerably the load on the service provider. Our system is based on a set of intelligent agents that are capable of requesting and offering services to one another.

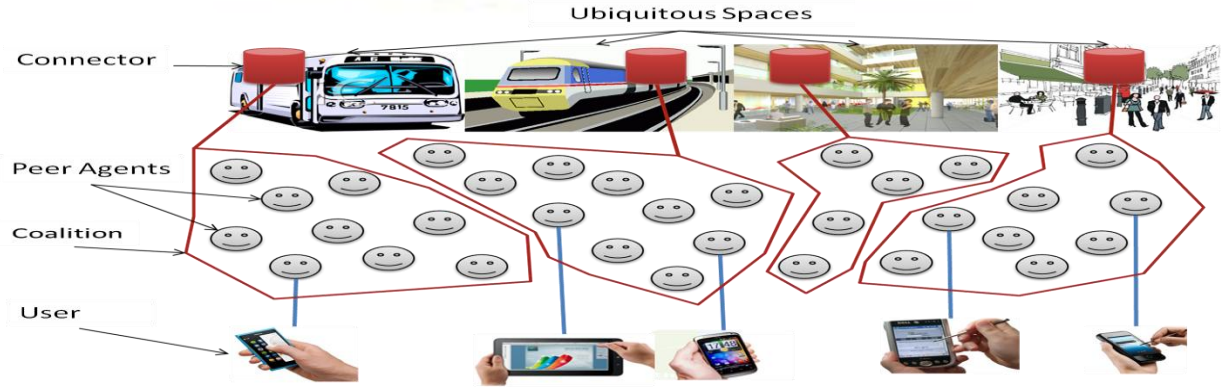

Figure 4. Static role based architecture

The system architecture is composed of nodes (UTs each one of them is hosting a peer agent) that are assembled in coalitions. Every ubiquitous space is equipped with nodes called connectors. Connectors can access the BDN by means of mobile agents that travels through BDN's nodes following a specific Workplans. These Workplans are generated using an evolutionary approach that optimizes criteria such as response delay, cost and quantity of collected data. In Figure 4 we provide a static role-based architecture (SRBA), peer agents are simple clients that requests services from connectors.

Connector architecture. As explained above, the connector is responsible for connecting users and providing routing information. When a user ask for a given service, the system need to offer that service in the best delay and with an optimized cost and quality of service. Therefore, we propose to cluster a number of agents inside the connector.

These agents are responsible for the optimization operations. Connector's architecture is based on the following agents (Figure 5):

- Identifier Agent (IA): These agents manage the decomposition of the requests which were formulated through a same short period of time ε* (ε-simultaneous requests). The decomposition process generates a set of sub-requests corresponding, for example, to sub-routes or to well-known geographical zones. Sub-requests are elementary independent tasks to be performed by the available set of distributed nodes (information providers) in the BDN. Each node must login to the system registering all proposed services. A service corresponds to the response to a defined task with fixed cost, processing time and data size. Therefore, an agent IA decomposes the set of simultaneous available requests into a set of independent tasks recognizing possible similarities, in order to avoid a redundant data research. The decomposition process occurs during the information providers' identification. Finally, the agent IA transmits cyclically all generated data to available scheduler agents. These ones must optimize the choice of the BDN nodes, taking into account some system constraints.
- Optimization Agent (OA): this agent checks if there are any similarities between user requests, thereby achieving the first step optimization and avoiding redundant searches. For example, user1 is asking for services {s_2, s_9, s_6, s_5} and user2 is asking for {s_1, s_2, s_5}. IA identifies that services S_2 and S_5 are requested by the two users, so there is no need to seek these services twice.
- Collector agent (CA): this agent is a mobile code that migrates to the destination nodes defined within the Workpan and collects the needed data. A fixed amount of data could be handled by every CA; thereby the number of CA is proportional to the total amount of needed data.
- Final solution agent (FSA): this agent is responsible for assembling final solutions for simultaneous requests. A response is ready to be sent to the user if all of the requested services composing the user's query are caught. There is no response if the deadline is obsolete and one or many services are not caught yet. The final result will be sent to the corresponding peerAgent.

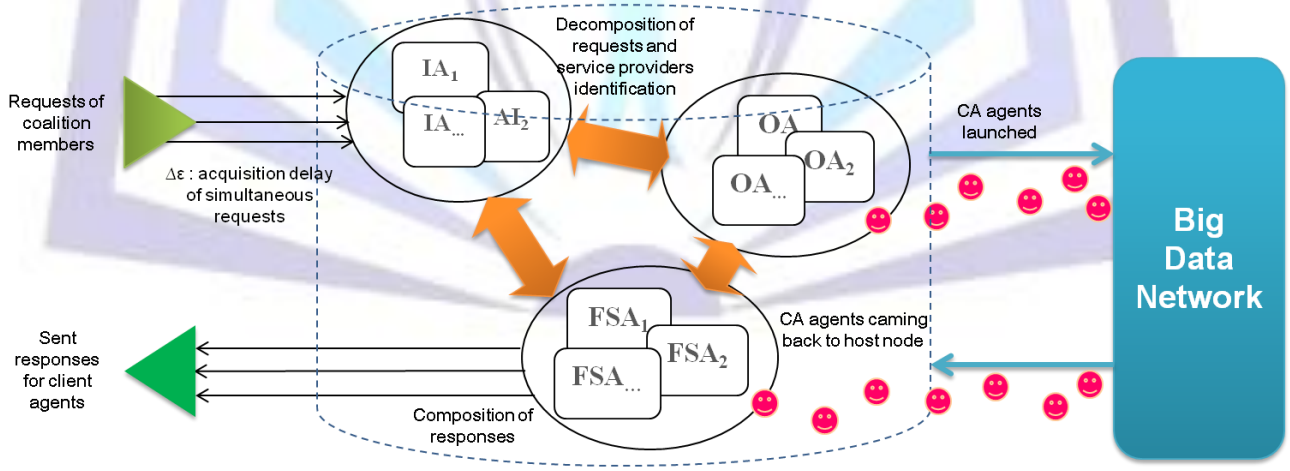

Figure 5. Connector architecture

We consider two types of service providers: permanent providers (PPs) and casual providers (CPs). PPs might be content providers or services providers that form the network with which CAs interact. PPs are registered within the system and supposed to offer data for a large period of time. On the other hand, CPs are practically users connected to the system who might share some services with each other in a P2P fashion. Incoming simultaneous client requests are treated at the first time by the IA which will identify any similarities between client queries by decomposing all queries to a set of elementary services. After that, it transfers the set of requested services to the OA which is responsible for the optimization of the assignment of services to PPs. Once the assignment table is ready, Workplans are established and assigned to CAs. CAs move to PPs nodes and interacts locally with them, needed data are collected and transmitted to

FSA once they get to the departure node (connector's node). FSA reconstructs response queries and sends them to correspondent clients.

Structure of peerAgent. This agent is hosted by user's UT. Within a temporary coalition formed with other peer agents, a peer agent should sometimes cooperate and sometimes negotiate in order to accomplish its objectives. This has to be done in real time. Therefore the time spent to reason about situation (perceiving the environment, choosing the best action and doing it) need to be minimal as much as possible. Thus, we adopt a belief-desire-intention (BDI) model and we define the agent's structure inspired by the Procedural Reasoning System (PRS) [9]. Referring to the BDI model, peer agent's beliefs in this case are mainly information that the agent knows about its host (autonomy status, storage capacity, etc.), about its owner (preferences, availability, etc.) and about other agents (their objectives). These data may change during runtime. Peer agent desires are defined by the objective function. The agent may execute some search tasks for particular services on behalf of its owner. PeerAgent intentions are actions that it decides to take considering a given state. Actions are pre-prepared and stored in so-called plans.

3.2. Distributed service search and composition approach

Workplan building using an evolutionary solution. In order to satisfy user requests, the system should take into account several criteria, namely response delay, information cost and quality of service. This can be formulated as a combinatory multi-objective problem. The target of this type of problem is to find the single solution giving the best compromise between multiple objectives.Previous works have proven the efficiency of evolutionary algorithms to rich optimal solutions using a suitable evaluation function. The search and the composition of distributed transport services are done by an evolutionary algorithm which is controlled by the current active OA agents. The selection of an appropriate representational scheme of a solution is fundamental to the success of evolutionary algorithms applications. In a previous work [26] [27] [28] [29], we designed an efficient coding for the chromosome (the solution) respecting the problem constraints. Therefore, we propose a flexible representation of the chromosome called Flexible Tasks Assignment Representation (FeTAR). The chromosome is represented by a matrix CH(I'xJ') where rows represent independent tasks (services), composing globally simultaneous requests and columns represent identified distributed nodes (providers). Each element of the matrix specifies the assignment of a node Sj to the task Ti as follows:

$$CH[i,j] = \begin{cases} 1: \text{if Sj is assigned to Ti} \\ *: \text{if Sj may be assigned to Ti} \\ x: \text{if Sj cannot be assigned to Ti} \end{cases}$$

We notice that each task must be assigned, so we assume that each task must be performed at least by a node, selected from a set of nodes proposing the concerned service. Indeed, the assignment and the scheduling of all tasks to the distributed servers represent the services composition. These services are proposed to clients as the optimal responses to their requests. The mobile agent Workplan problem can be described as follows: CA agents are created and initially launched from an originally node (Home node). The other network nodes represent available information providers where an CA agent can move to collect data corresponding to claimed services. The same service can be proposed by different nodes with different cost, processing time and different quality. On Home node, processing time is null. We call the response time of a service on a node, the processing time of the correspondent task to this service on this node. Our goal is to minimize CA agent number and their navigation time in order to explore all the BDN, taking into account network state. We introduce some definitions using variables described in table 2 below:

Definition 1: CT_i (Processing time on node F_i) is the needed computing time on the node F_i to extract the data quantity Qt_i.

Definition 2: Qt_i (Data quantity on node F_i) is the average data size to extract from F_i.

s_i represents the elementary service proposed by F_j. Therefore:

$$Qt_j = \frac{\sum_{i=1}^{I} a_{ij} Q_i^j}{\sum_{i=1}^{I} a_{ij}} \quad (1) \qquad CT_j = \frac{\sum_{i=1}^{I} a_{i,j} P_i^j}{\sum_{i=1}^{I} a_{i,j}} \quad (2)$$

Where a_{ij} is a Boolean value as follows: $a_{ij}=1$ if the node F_j propose a service for the task T_i and $a_{ij}=0$ otherwise (according to the given service table).

Table 2. Notations.

Variable	Description
m	Number of CA agents
$CA_1,\ldots,CA_m$	Identifiers of CA agents
H	Home node
Wk_i	Nodes sequence representing an CA_i agent Workplan: $(F_{i1},\ldots,F_{ip})$ with $1\leq p\leq J$
$T(Wk_i)$	Routing time for Wk_i
$Qte_{k,u}$	Transported data size until node F_u of an agent CA_k
$Tr(Qte_{k,u},F_u,F_v)$	Transmission time for $Qte_{k,u}$ from node F_u to node F_v

CT_i	Processing time on node F_i
Qt_i	Data quantity on node F_i
$d(F_i,F_j)$	Data transfer rate between nodes F_i and F_j

Definition 3: $Qte_{k,u}$ (Data quantity transported until F_u by CA_k) is the collected data quantity by CA_k during its route, until node F_u.

$Qte_{k,u}$ is computed like this: $Qte_{k,u} = Q_0 + \sum_{r=1}^{u} Qt_{kr}$ (3)

Definition 4: $Tr(Qte_{k,u},F_u,F_v)$ (Transmission time) needed time for the agent CA_k to migrate from F_u to F_v transporting the data quantity $Qte_{k,u}$.

$Tr(Qte_{k,u},F_u,F_v)$ is computed like this: $Tr(Qte_{k,u},F_u,F_v) = \frac{Qte_{k,u}}{d(F_u,F_v)}$ (4)

Definition 5 : $T(Wk_k)$ (Routing time for Wk_k) is the needed time for the agent CA_k to visit the sequence of network nodes $(F_{k1},\ldots,F_{kp})$ with $1 \le p \le J$.

$T(Wk_k)$ is computed like this : $T(Wk_k) = T_{go} + T_{travel} + T_{return}$ (5)

Table 3. Total transmission delay.

	Wk_k	Tgo	T_{return}	T_{travel}
p=1	(F_{k1})	$Tr(Q_0,H,F_{k1})$	$Tr(Qte_{k,p},F_{kp},H)$	CT_{k1}
1<p≤J	$(F_{k1},\ldots,F_{kp})$			X_k

With: $X_k = \sum_{i=1}^{p} CT_{ki} + \sum_{i=1}^{p-1} Tr(Qte_{k,i},F_{ki},F_{ki+1})$ (6)

To propose a cost-effective Workplan mobile agent scheme, we assume the existence of a monitoring module which provides information about network status (latency, bandwidth, traffic, bottleneck, failure…). Therefore, we can get data transfer rate values among all pairs of nodes through the BDN. The goal is to find a set of CA agents minimizing their navigation time in order to explore all the BDN nodes, taking into account network state. It is clear that sending an CA to each node gives us the best total computing time because, in this case, agents are launched simultaneously into each network node. We keep this best total computation time to build nodes partitions, minimizing the number of CA agents. Consequently, we just care about data cost and processing time in the service table (table 1), ignoring data cost. As described previously, $d(F_i,F_j)$ namely data transfer rate among two network nodes F_i and F_j, is available. We give here a brief description of the algorithm detailed in [26]:

The initial Workplan algorithm description

Step 1: sort nodes in decreasing order according to their correspondent routing time $T(Wk_i=F_i)$ $\forall 1 \le i \le J$. Set the threshold δ which is the routing time of the first node in the sorted list: $\delta = \max_{1 \le i \le J}(T(Wk_i = F_i))$ (7)

Step 2: partition the given network into several parts by gathering nodes so that the routing time of each part does not exceed the threshold δ.

According to a generated FeTAR instance CH, equations (1) and (2) become:

$$Qt_{cj} = \sum_{i=1}^{I'} (a_{c_i c_j} \times Q_{c_i}^{c_j}) \quad (1') \qquad CT_{cj} = \sum_{i=1}^{I'} (a_{c_i c_j} \times P_{c_i}^{c_j}) \quad (2')$$

With $a_{c_i c_j}$ a Boolean value as follows: if $CH[c_i,c_j]=1$ $(1 \le i \le I'$ and $1 \le j \le J')$ then $a_{c_i c_j}=1$ else $a_{c_i c_j}=0$. Total transmission time of the final Workplans are deduced using equations (1'), (2'), (3), (4), (5) and (6) according to table 3.

3.4. Coalition Overview

Aim of coalition formation. Mobility is a fundamental element in people's daily activities. Because users are always moving (house, bus, train, office, supermarket, road, park, sport clubs, etc.), it is important to take account of this dynamic aspect when we design an information system architecture. Users may be considered as mobile nodes that could be available a certain period of time in a specific geographical area. For example, certain number of users is gathered during a train trip, agents' coalition in this case is the technological representation of the group of users. The purpose of the coalition formation is to provide a more performing information system. Users collaborate in order to provide a more flexible and reliable service search and storage tasks. Our proposition consists of giving the ability to the system users to switch their role from simple clients to service providers of services that they requested from the connector. We distinguish

two types of agents: passive agents and active agents. Passive agents are simple client agents that request services without any contribution to the optimization process of the system. Active agents are agents that are endowed with service providing capability and are envolved in load minimization upon original service providers. In the Figure 6 we provide the dynamic role based architecture: passive agents are represented by normal smileys and active ones are represented by hatted smileys.

Service distribution and search strategy. This task is handled by the connector in the beginning. In fact agents do not know anything about service providers. They request the connector to provide the suitable services. When the number of connected agents surpasses a defined threshold, the connector change its behavior, it becomes capable of managing services inside this new coalition of agents. A number of relevant services are distributed among the coalition and agents don't need any more the connector to optimize the search task, they are capable of requesting optimized services directly from each other. Currenly we are using the Yellow pages strategy for implementing service publishing and registration. However other techniques such as DHT may be applied.

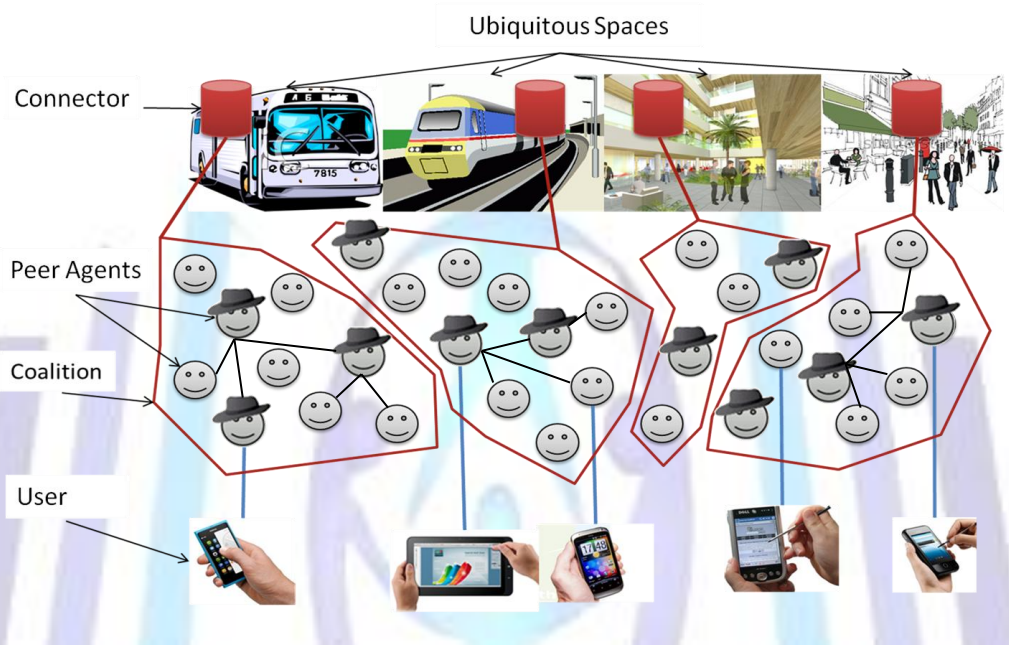

Figure 6. Dynamic role based architecture.

Role switching. One of the most important aspects of our model is the role switching. In fact, conventional information systems consider users as clients without considering the possibility to integrate some server functionalities within the front end application. In our system, user agents are all capable of fulfilling client role (requesting services, displaying results, etc.) and in the same time hosting some services for the benefit of the coalition. A dynamic policy of role switching is fixed by the connector. Indeed, some agents are allowed to switch the role and became "active agents", others are not. It depends on the redundancy rate and the number of the coalition members.

The role switching strategy is proposed in order to maintain the system robustness and reliability by reducing the requests load upon original service providers. In the next section we present some results that support our proposition.

4. SIMULATION RESULTS AND ANALYSIS

The proposed system architecture is composed of two principle components: the connector and the agents' coalition. These two main subsystems are working with each other in order to maintain high services availability, real time query response and an optimized service discovery and search.

In this section we present the simulation results of the role switching strategy in the coalition subsystem. The connector is simulated in previous works [26] and replaced in this work by a cluster of service providers. Network transmission is implementend using the model of the section III.2. We are using JADE (Java Agent Development) framework to create our agents. JADE is a middleware for the development and the execution of real time agent-oriented applications. It implements FIPA[6] (Foundation for Intelligent Physical Agents) specifications and uses the FIPA standard communication langage ACL (Agent Communication Langage) which enables interoperability through FIPA compliance. It implements also the agent mobility.JADE has a library called LEAP dedicated for the application development for mobile and limited resources devices (ANDROID, PJAVE and MIDP). For the supervision of the platform, JADE offers a set of graphical tools such as the remote management agent (rma) illustrated in Figure 7, the dummy agent, sniffer agent, etc.

[6] http://www.fipa.org/

Using this platform we programmed the coalition agents. Many scenarios are conducted. The main goal of the simulation is to provide a comparaision between the SRBA and the DRBA. We test the performance of the hole system relative to response time of client agents before and after the role switching. We provide also statistics about the requests load upon active agents. First of all, the simulation is composed of stationary servers that represents the original service providers. Connectors are hosting coalition members (clients) who are mobile agents that are in a continuous move from a connector to an other. Clients are recording the response time by sending requests and waiting for responses and many other information. Then, these collected data are sent to a GUI agent that is updating its GUI each 100 milliseconds (Figure 8). We are using tolens to represent agents' states: "♣" for passive agents which still simple clients and both "♣ ♦" for active agents which still clients and have acquired the role switching ability.

The data GUI can give us a real-time information about agent locations, number of the sent requests and received responses. It is important to clarify that we have modeled the loss probability in a wireless network. This means that not all of the sent requests arrive to the destination, there is a probability that a request is lost because of network uncertainty (availability, mobility, limited bandwidth, etc.).

We are using 10 agents which are moving from a connector to an other and sending requests. In the first test illustrated in Figure 9, all agents are passive agents (clients) and do not change their role all along the runtime. After the transition phase, the mean response time (MRT) is constant and roughly equal to 2 seconds.

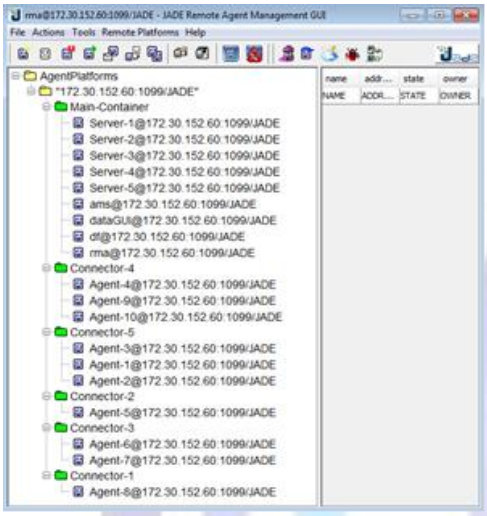

Figure 7. JADE rma agent GUI.

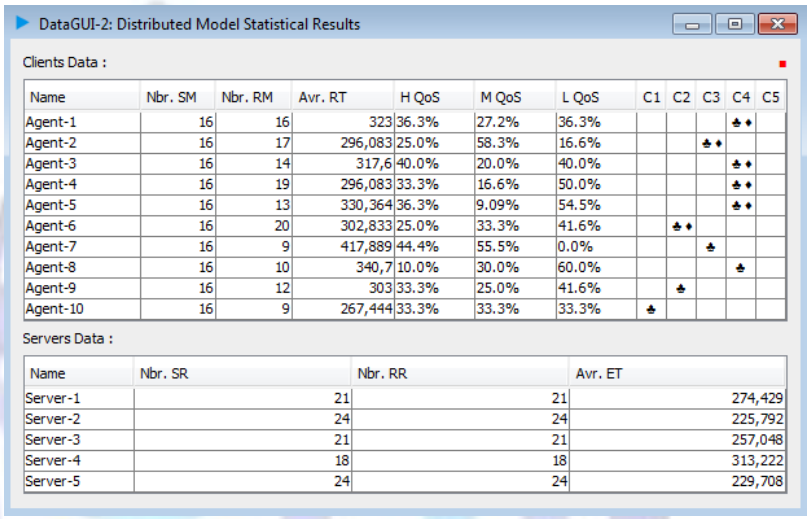

Figure 8. Data GUI agent.

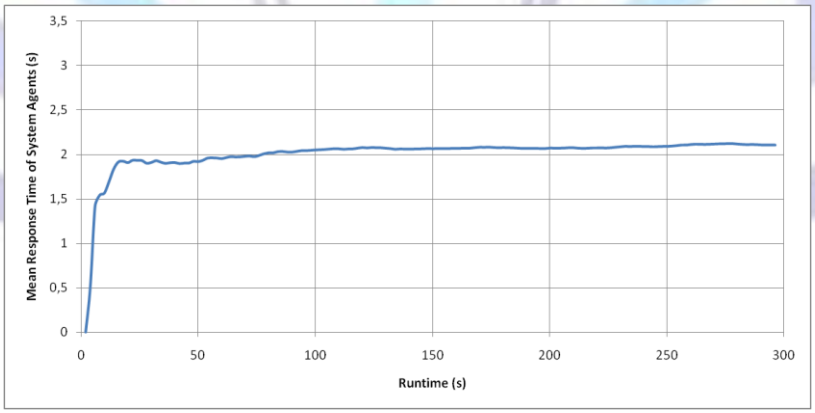

Figure 9. MRT before applying role switching strategy.

However, after applying the role switching startegy we can see in Figures 10 and 11 that the MRT decreases progressively until it reaches 0.5 seconds. In fact, systems agents starts requesting services from original service providers. After a 40 seconds of runtime some agents will switch their role and become also service providers. From this moment, the response time of the hole systems starts. When we increase the number of active agents we get roughly the same results.

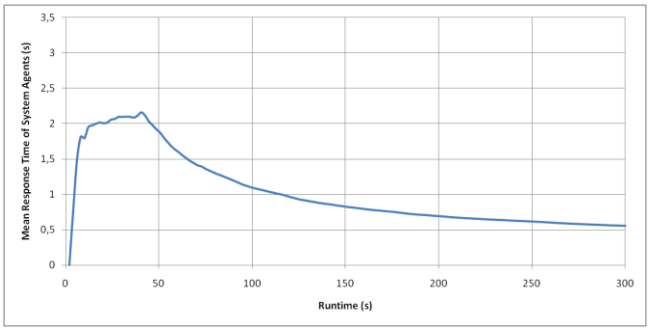

Figure 10. MRT of peer agents after 2 agents switch their role.

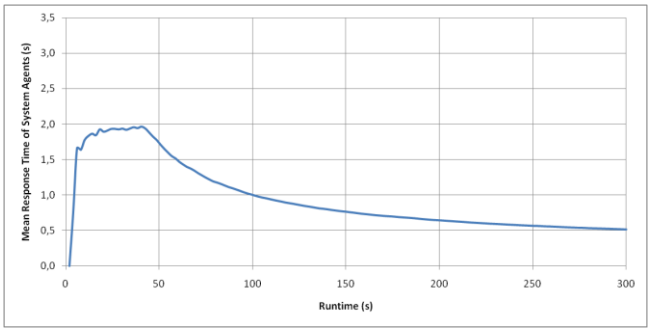

Figure 11. MRT of peer agents after 4 agents switch their role.

In Figure 12 we present the MRT of all agents at the end of the runtime (around 0.5 seconds). In fact the response time is a function of the UT performance and the network status. Cases in Figure 12 and 13 corresponds to scenarios where the number of active agents is respectively equals to 2, 4, 6, 8 and 10.

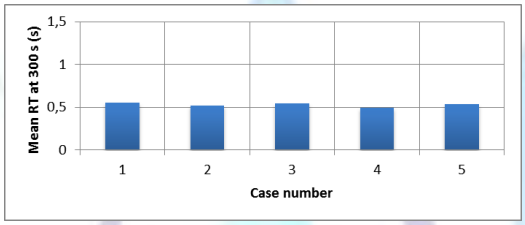

Figure 12. MRT at 300 s of simulation with variation of the number of the active agents. Case 1 : 2 active agents ; case 1 : 4 active agents ; case 1 : 6 active agents ; case 1 : 8 active agents ; case 1 : 10 active agents

Finally we present the system load in terms of received requests by active agents. More we have active agents less requests are processed by each one of them.

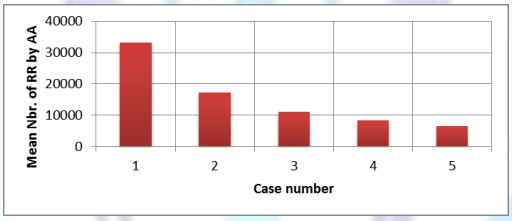

Figure 13. Mean number of received requests by active agents with variation of the number of active agents.

In table 4 we provide a summary of the comparaison between the static role-based architecture and the dynamic role-based architecture.

Table 4. Comparison between static and dynamic role-based architectures.

	Static role based architecture	**Dynamic role based architecture**
Clients behaviour	Allways request services	Request and provide services
Servers status	Allways reuqested	Requests decrease over time
System load for the same number of users	Constant	Decreases over time
Response delay for th same number of users	Constant	Decreases over time

5. CONCLUSION AND FUTURE WORK

We proposed an agent-based architecture for ubiquitous information system. The system is composed of connectors and coalitions. Connectors are responsible for receiving client requests and resolving a multi-objective optimization problem. Incoming simultaneous requests are decomposed into a set of elementary services where similarities between requests are spotted in order to reduce research redundancy. Then suitable service providers are assigned to services and CA Workplans are generated through an evolutionary optimization process. CA will navigate the BDN and collect the needed information. Collected data is distributed amoung coalition members. Then, we presented an innovative strategy for enabling our system to scale and to handle the high amout of simultaneous requests. In fact, clients are now endowed of role switching capability which enables them to became service providers. From this moment, services inside the coalition won't be searched another time while there is active agents providing these services. The experimental results show that the DRBA's MRT of a coation of agents is much better than SRBA's one. Besides, on the DBRA, the hole system load is

reduced. Over rutime original service providers receive less and less requests from users, and more clients are active less request are processed by each one of them.

In the future work we will focus on the improvement of our proposed architecture. The connectors will be conneted and the used techniques inside coalition will be applied to a larger scale where connectors will take the place of peer agents. We will also detail a partial agreement negociation protocol between peer agents. The negociation protocol will provide a more flexible communication support for coalition members. We are also thinking of testing our system on a real testbed such as SensLAB.

ACKNOWLEDGMENTS

This work is supported by the French Environment and Energy Management Agency (ADEME) and the Nord-Pas-de-Calais region under a PhD thesis finance contract.

REFERENCES

[1] Adam E., Mandiau R., “Flexible Roles in a Holonic Multi-Agent System”, Holonic and Multi-Agent Systems for Manufacturing. Volume 4659, 2007, pp 59-70.
[2] Adam E., “Modèle d’organisation multi-agent pour l’aide au travail coopératif dans les processus d’entreprise : application aux systèmes administratifs complexes”. Université de Valenciennes et du Hainaut-Cambrésis, PhD Thesis, 2000.
[3] Burin de Roziers, C.; Chelius, G.; Ducrocq, T.; Fleury, E.; Fraboulet, A.; Gallais, A.; Mitton, N.; Noel, T.; Vandaele, J., “Using SensLAB* as a First Class Scientific Tool for Large Scale Wireless Sensor Network Experiments”, Networking 2011. Volume 6640, 2011, pp 147-159.
[4] Durmus, A.; Erdogan, N., "A web services market framework with role based agents," Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on , vol., no., pp.4722,4727, 11-14 Oct. 2009.
[5] Depke R., Heckel R., Küster J. M., “Improving the Agent-Oriented Modeling Process by Roles”, in Proceedings of the Fifth International Conference on Autonomous Agents. Pages 640-647, 2001.
[6] Ferber, Jacques. Les Systèmes Multi Agents: vers une intelligence collective. Inter Editions, 1995.
[7] Ferrari, L.; Haibin Zhu, "Enabling dynamic roles for agents," Collaboration Technologies and Systems (CTS), 2011 International Conference on , vol., no., pp.500,507, 23-27 May 2011.
[8] Ferrari, L.; Haibin Zhu, "WhiteCat: Making agent roles perceivable," Collaborative Technologies and Systems (CTS), 2010 International Symposium on , vol., no., pp.637,638, 17-21 May 2010.
[9] Georgeff M., “Decision Making in an Embedded Reasoning System”. California, SRI International, 1990.
[10] Haibin Zhu; MengChu Zhou, "Issues in Adaptive Collaboration," Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on , vol., no., pp.2828,2833, 10-13 Oct. 2010.
[11] Haibin Zhu; Ming Hou; MengChu Zhou, "Establishing the foundation of adaptive collaboration," Collaborative Technologies and Systems (CTS), 2010 International Symposium on , vol., no., pp.546,554, 17-21 May 2010.
[12] Haibin Zhu; MengChu Zhou, "Roles in Information Systems: A Survey," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on , vol.38, no.3, pp.377,396, May 2008.
[13] Haibin Zhu; MengChu Zhou, "Role-based collaboration and its kernel mechanisms," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on , vol.36, no.4, pp.578,589, July 2006.
[14] Jennings, N., et M. Wooldridge. «Software agents.» IEE Review, 1996: vol.42, no.1, pp.17-20.
[15] Ohashi M., “Japanese Ubiquotous Network Project: Ubila”, Handbook of Ambient Intelligence and Smart Environments, 2010, pp 1229-1255.
[16] Pouwelse, J. A.; Garbacki, P.; Epema, D. H. J.; Sips, H. J., “The Bittorrent P2P File-Sharing System : Measurements and Analysis”, Peer-to-Peer Systems, 2005.
[17] Roussopoulos, M., M. Baker, D. Rosenthal, T. Guili, P. Maniatis, et J. Mogul. «2 P2P of Not 2 P2P?» International workshop on Peer-To-Peer Systems, IPTPS 2004, 2004.
[18] SensLAB (http://www.senslab.info/).
[19] Stajano F., “Security Issues in Ubiquitous Computing”, Handbook of Ambient Intelligence and Smart Environments, 2010, pp 281-314.
[20] Stoica I., Morris R., Karger D., Kaashoek M. F., Balakrishnan H., “Chord: a scalable peer-to-peer lookup protocol for internet applications”, IEEE/ACM Transactions on Networking SIGCOMM'01, 17-32.
[21] Weiser M. "Hot topics-ubiquitous computing", Computer, Oct 1993, vol.26, no.10, pp.71-72.
[22] Wooldridge M., Jennings N. R., and Kinny D., “The Gaia Methodology For Agent-Oriented Analysis And Design”. Journal of Autonomous Agents and Multi-Agent Systems. 2000.
[23] Wooldridge M., An Introduction to MultiAgent Systems. John Wiley and Sons, 2002.
[24] Xiao L., “An adaptive security model using agent-oriented MDA”, Information and Software Technology, Volume 51, Issue 5, May 2009, Pages 933-955.
[25] Yan Q., Shan L., Mao X., Zhi-chang Qi, “RoMAS: A Role-Based Modeling Method For Multi-Agent System”, International Conference on Active Media Technology 2003.
[26] Zgaya H., Hammadi S., Ghédira K. “Workplan Mobile Agent for the Transport Network Application”, IMACS’2005, Paris 11-15 July 2005, pp61.
[27] Zgaya, H.; Hammadi, S., "Dynamic approach to reassign tasks when servers breakdown in a Multimodal Information System," Computational Engineering in Systems Applications, IMACS Multiconference on , vol.1, no., pp.985,991, 4-6 Oct. 2006.

[28] Zgaya, H.; Hammadi, S., "Transport Services System Integration and Optimization in Agent Based Model," Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on , vol., no., pp.29,29, Nov. 28 2006-Dec. 1 2006.

[29] Zgaya, H.; Hammadi, S., "Assignment and Integration of Distributed Transport Services in Agent-Based Architecture," Intelligent Agent Technology, 2006. IAT '06. IEEE/WIC/ACM International Conference on , vol., no., pp.96,102, 18-22 Dec. 2006.

Author's Biography

Ayoub Bousselmi

Ayoub Bousselmi is actually a PhD student within LAGIS laboratory in French high school Ecole Centrale de Lille. His current reaserch is at the intersection of information systems, distributed optimization and distributed artificial intelligence. Born in Tunis (Tunisia) in 1987. He received the Ingineer degree in Electrical and Electronic from the National Engineering School of Gabes (Tunisia) in June 2011.

Dr. Hayfa Zgaya

Dr. Hayfa Zgaya is an Associate Professor of Logistics and Health Informatics at the Institute of Engineering in Health of Lille (France). She obtained her PhD degree in July 2007 from the high French school "Ecole Centrale de Lille" and her master degree in November 2002 from the French Polytechnic High School of Nantes University. Her main research areas are the optimization, the artificial intelligence and the logistics issues. She is member of the Public Health department EA 2694, University of Lille 2.

Thomas Bourdeaud'huy

Thomas Bourdeaud'huy is assistant- professor at Ecole Centrale de Lille, member of the logistics systems optimization team in lagis laboratory, Lille.
He has received his phd in 2004. He is leading the chair of e-business at ecole centrale de Lille and co-animates the french working group on Petri nets.

Mr. Slim Hammadi

Mr. Slim Hammadi is a full Professor of production planning and control at the Ecole Centrale de Lille. Born in Gafsa (Tunisia) in 1962, he has obtained by 1988 the Master degree in Computer science from the University of Lille (France). Pr Hammadi obtained a PhD degree in job-shop scheduling and control in 1991 at Ecole Centrale de Lille. He is a senior member of IEEE/SMC and has served as a referee for numerous journals including the IEEE Transactions on SMC. Pr. S. Hammadi was Co-Organizer of a Symposium (IMS) of the IMACS/IEEE SMC Multi conference CESA'98 held in Hammamet (Tunisia) in April 1998. He has organized several invited sessions in different SMC conferences where he was session chairman. He was chairman of the International congress on "Logistic and Transport" LT'04, MHOSI'05, LT'06 and LT'07. His teaching and research interests focus on the areas of production control, production planning, computer science, discrete and dynamic programming and computer integrated manufacturing.