uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Hamid Harroud

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

An Agent-based Framework for Building Mobile Applications

TITRE DE LA THÈSE / TITLE OF THESIS

Ahmed Karmouch

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Sivarama Dandamudi

Amiya Nayak

Samuel Pierre

Tet Yeap

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

An Agent-based Framework for Building Mobile Applications

By

Hamid Harroud

A thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

in partial fulfillment of the requirements for the degree of

Doctorate of Philosophy

in

Computer Science

Supervisor

Dr. Ahmed Karmouch

Ottawa-Carleton Institute for Computer Science

School of Information Technology and Engineering

University of Ottawa

Your file Votre référence
ISBN: 978-0-494-18109-6
Our file Notre référence
ISBN: 978-0-494-18109-6

Abstract

Recent advances in portable computing devices and wireless technology make it possible to provide seamless and ubiquitous services for mobile users. As users move from one location to another, there is a greater need to automatically provide office-like environments that normally have to be configured and setup on an individual basis by each user at each visited location. Due to occasional disconnections, device capabilities and network heterogeneity, users' mobility places a considerable burden on applications to be used by mobile users. On the other hand, it receives little assistance from existing infrastructure and Internet protocols for tracing users' movement and adapting services to satisfy unique characteristics of mobility.

This thesis recognizes the importance of managing users' mobility at different levels. From the network level by monitoring network devices and access points to the organizational level by assigning a personalised profile and a specific role to mobile users at visited locations. We propose a set of policy-based agents composing a framework that assists applications and services in dealing with mobility challenges and capable of providing a mobile working environment, similar to the home or office environment, to be used by users wherever they travel and whichever devices they use. The framework enables application designers to more easily build complex mobile and context aware applications. It also provides some basic services so that it can help mobile users simply by making use of the existing support and built in policy-based software agents.

In addition, three different applications built upon the framework along with various scenarios have been discussed. Various scenarios illustrate how the framework can be helpful in providing support and services to mobile users. The framework is also a test-bed platform that can be used for the ongoing and future research projects that make use of mobility and context-awareness in MMARL laboratory.

Acknowledgements

I would like to thank my supervisor Professor Dr. Ahmed Karmouch whose vision and enthusiasm has had a great impact on my scientific development and the research work presented in this thesis. He formulated the initial idea of managing personal mobility in the Mobile Alliance Project and hired me to work on this excellent topic. The prototypes presented in this thesis work would not have been possible without his generous expenditure on the lab equipment to help me and all the students implement and better investigate various research issues. Professor Karmouch was instrumental in shaping the contents of this thesis by encouraging me to publish in conferences and journals, and making numerous corrections to the submitted and accepted publications. Without his constant encouragement and patience during my difficult times this thesis would not have been possible. In many ways he not only helped shape my research work, but also future research direction, my academic and personal skills. He is truly a great advisor and I am truly grateful to him.

I am grateful to all my colleagues in the laboratory for their co-operation and fruitful discussions. I really enjoyed the working environment and the very positive atmosphere at the MMARL lab. I am particularly indebted to colleagues that have been involved in the prototypes implementations.

I would also like to thank my Ph.D. committee members for accepting to examine and evaluate the dissertation work.

Producing a PhD really needs sacrifices which are rarely made alone. I would like to thank my wife, Keltoum, for all the sacrifices she has been forced to make (and still makes) for my research work. I strongly appreciate her endless patience and support throughout all these years; and also my kids, Adil, Mounia and Zaineb, who were a source of joy and great motivating factor when I was tired. I am of course grateful to my parents for their many sacrifices that made my education possible. They were a constant source of encouragement during my studies and research.

Special thanks go to the Faculty of Graduate and Postdoctoral studies at the

University of Ottawa for their financial support throughout my Ph.D. program, the Natural Sciences and Engineering Research Council of Canada NSERC research funding, Mitel Corporation, Nortel Networks, the National Research Council (NRC) of Canada and all those that have offered their support and encouragement whilst working on this thesis.

Table of Contents

List of Figures

List of Tables

LIST OF SYMBOLS AND ABBREVIATIONS

FIPA	Foundation for Intelligent Physical Agents
IETF	Internet Engineering Task Force
PDA	Personal Digital Assistant
FIPA-OS	FIPA Open Source (Agent Platform)
ACL	Agent Communication Language
SIP	Session Initiation Protocol
IETF	Internet Engineering Task Force
COPS	Common Open Policy Service
RDF	Resource Description Framework
CC/PP	Composite Capabilities/Preference Profiles
XML	eXtensible Markup Language
PMA	Policy Management Agent
PIB	Policy Information Base
PSA	Policy Service Agent
CLNP	Context Level Negotiation Protocol
CMA	Context Management Agent
PA	Personal Assistant
UA	User Agent
NSA	Network Service Agent
W-SIP	Wrapper SIP User Agent
SDP	Session Description Protocol
SA	Service Agent
MMA	Mobility Management Agent
TA	Team Assistant
OWL	Ontology Web Language
UID	Unique IDentification number (TagID)
PAN	Personal Area Network

CHAPTER 1

INTRODUCTION

1.1. Motivation

The proliferation of small sized devices and notebook computers coupled with globalization of markets and business processes has led to an increasingly nomadic computing lifestyle.

Since personal computing devices are portable and Internet access is becoming ubiquitous, today's increasingly mobile users are now requiring personalized services at different locations on their network connections and hardware devices. Users moving outside their office expect maintaining an office-like environment at home, at temporary locations such as a meeting at another company, while in a business trip or in a hotel. Locations that a user may visit can be configured into a private network in

which users can retain access to services provided by their home site or locally by the visited site as they move from one location to another.

Figure 1 presents an example of mobile computing environment. A mobile user may interrupt his work and leaves the office to attend a meeting at a local or remote hotel and then prepares the meeting report when at home. The user could be requested on one of his devices (PDA, Phone, Laptop) while on his way either to the hotel or home.

While users' mobility places a considerable burden on applications that communicate

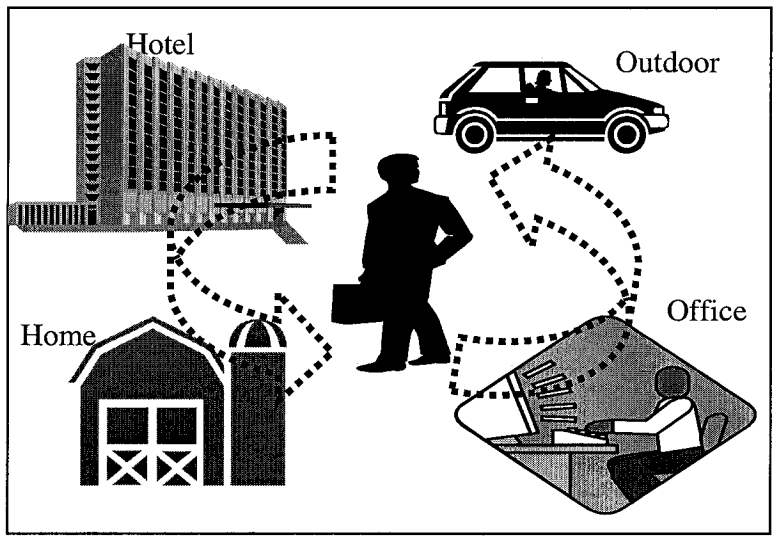

Figure 1. Generic scenario for a mobile user

across the network due to occasional disconnections, devices capabilities and networks heterogeneity, it receives little assistance from existing infrastructure and Internet protocols for tracing users' movement and adapting services to satisfy unique characteristics of mobility.

This thesis recognizes the importance of managing users' mobility at different levels. From the network level by monitoring network devices and access points to the organizational level by assigning a personalised profile and a specific role to mobile users at visited sites. We define a set of system components to assist services in dealing with these challenges and capable of providing a mobile working environment, similar to the home or office environment, to be used by users wherever they travel and whichever devices they use.

We propose a solution based on the use of authorization policies and agents

negotiation. Authorization policies are defined at each site to monitor resource utilization as well as the site's overall policies. The use of policies enables dynamic changes in the behavior of a system without altering its components. The agent negotiation process is used to guide the interaction of agents as they make decisions on users' behalf in accordance with established policies. For example, when a user moves from a home site to a visited site, system agents communicate and negotiate with one another to allocate a temporary profile to the user according to his/her home preferences and services. The profile includes elements such as the user's identity, the authorized services and their associated quality of service, the devices that may be used, and possibly costs limits to be paid. Mobile users may wish to use various devices with different capabilities, such as a workstation, a laptop with a wireless connection, a PDA or a phone. The system therefore needs to be able to identify the device in use at any particular time and to adapt the service presentation to the device's capabilities.

The interest in agent technology to support mobility is mainly motivated by intrinsic properties such as its autonomy and its ability to migrate. This allows a temporary disconnection between the end device and the network, while agents continue to perform their tasks. They return results when the mobile user reconnects. This is crucial, as mobile users often have to work in unstable network conditions, especially with wireless connections.

When an agent needs to interact with a remote service or user, it packs its code and its state variables, sends them to the remote location, and then resumes its activities at that location. However, it is necessary that an agent, migrating from one location to another, has the capability of perceiving the environment in which it executes and responds to the changes that occur in that environment. The use of policies enables to achieve dynamic changes in the behavior of the system agents in a flexible manner. We propose to apply policies through several levels of the system including security, admission control, user profile, resource management, and service personalization.

We propose also to attach policies that are related to an agent's mission to that agent. By integrating policies with an agent, they become in effect part of the agent's model.

Contextual information that characterizes the environment in which agents execute could be captured and translated to a set of context policies. The policy translation and enforcement are therefore tightly bound to the context in which it is evaluated. Based on the agent's state, tasks or services and resources conditions, a policy will be triggered, forcing the agent to adapt accordingly.

1.2. Domains and Impact of Mobility

As shown in figure 2, mobility can take the form of users moving between fixed terminals anywhere at different sites (personal mobility) or users taking mobile devices with them wherever they move (terminal mobility) or a user maintaining a session while switching between multiple devices (session mobility) or service mobility. In all scenarios, the mobility should support seamless movement of users and/or services between different locations by addressing the dynamic changes that may occur when at visited sites.

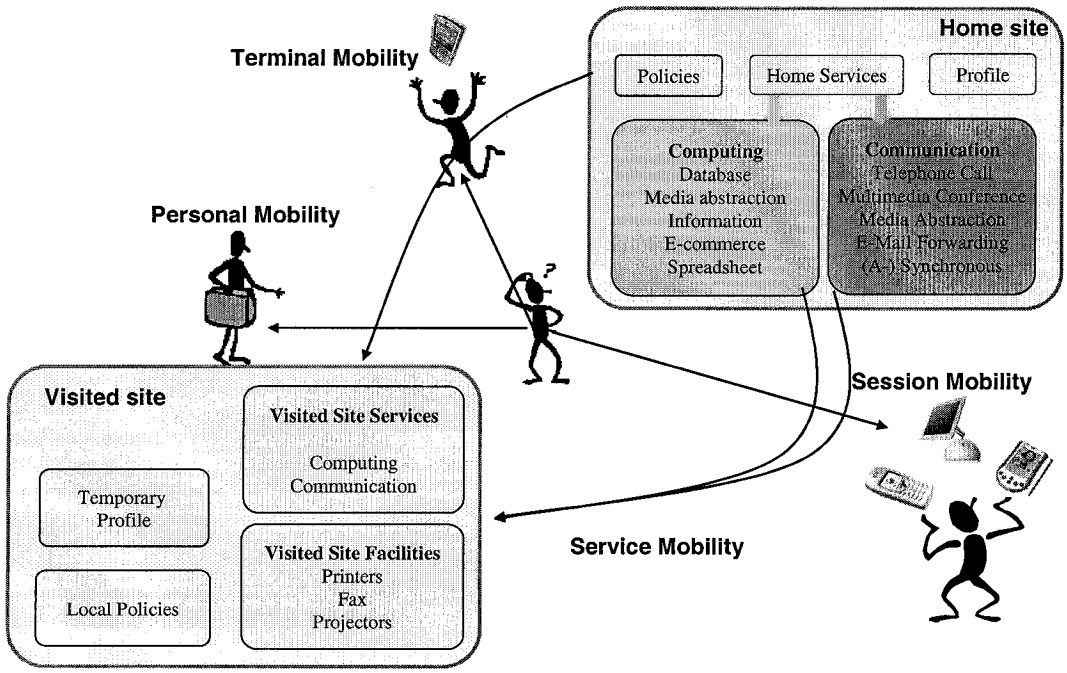

Figure 2. Mobility Types

A unified global vision of mobility has been specified by the IMT-2000 initiative [4]. Some of the important goals of IMT-2000 involve global roaming across heterogeneous networks and support of high-speed multimedia sessions in a mobile environment. The four mobility aspects (i.e. personal, terminal, service and session mobility) are considered in the following subsections.

1.2.1. Personal Mobility

Personal Mobility allows a user to access services independently of terminals and networks. Users are not tied to their terminals. They can travel, find suitable terminals and network access points and start services. The most important requirement in such kind of mobility is adaptation. Services should be adaptive to different terminals that could be used by mobile users, adaptive to different transmission links, and adaptive to different contexts of use.

1.2.2. Terminal Mobility

Terminal Mobility is defined as the possibility to the terminal to change location while maintaining the service active. It permits users to move with their terminals. Users do not need to terminate the service, then to move and finally to restart the service.

1.2.3. Service Mobility

Service Mobility is defined as the ability of the network to provide personalized services to mobile users. It permits users to maintain access to their services while moving or changing devices [2]. Two aspects must be considered: (i) maintain an adequate QoS for the duration of the session, (ii) ensure that users have access to all their subscribed services regardless of the point of attachment to the network [100].

1.2.4. Session Mobility

Session Mobility is defined as the ability of a user to maintain an active session while switching between different devices. For example, a user may need to continue a session that has been started on a mobile device on his/her workstation when he/she enters the office. Parts of a session could also be moved to specialized devices for audio and/or video such as a video projector.

1.2.5. Impact of Mobility

Mobility is not a service itself, but enhances the availability of other services. It is a capability that can be added to any communication system. Mobility has an impact on mobile communication systems on all the components, including devices, networks and services.

There are currently a wide variety of wireless network services available (i.e. IEEE 802.11, CDPD, GPRS, CDMA, Bluetooth, etc.) and different device types (i.e. PDAs, Laptops, Smart phones, Tablets, etc.) with different capabilities and multiple operating systems (i.e. PocketPC, Palm, Windows 2000/CE/XP, RIM, etc.), together with advances in computing technology have brought more miscellaneous services to be delivered with various qualities.

To expand the roaming range through heterogeneous wireless networks, an end-user device may require multiple network interfaces. As an example, consider the case of a user who moves from an indoor wireless LAN (IEEE 802.11) to outdoor wide-area data network (CDPD). Due to the disparities in multiple parameters such as modulation schemes, frequency bands and data rates, network interface cards need to be switched for indoor and outdoor wireless transmission. Furthermore, heterogeneity is encountered across data and telephony wireless networks. For example, heterogeneity prevails when a user on the Internet aspires to connect to a user on Public Switched Telephone Network (PSTN).

A global network would allow seamless integration of terminals, networks, and mobile applications, so that they can be smoothly accessed with various devices across a wide range of wire-line and wireless architectures. Establishment of a global, integrated telecommunications and data network is being actively deliberated in Europe as Universal Mobile Telecommunications System (UMTS) [3] and in North America as International Mobile Telecommunications 2000 (IMT-2000) [4].

The trend in the growing popularity of mobile computing technologies appears to continue. The diagram below (figure 3) illustrates that there are approximately 100 million mobile and remote workers today, rising to over 160 million by 2006 in the world [5].

Ideally, mobility should be completely transparent to users. Transparency relieves users of the need to be constantly aware of the details of their computing environment, thus allowing them to focus on the real tasks at hand. The adaptation necessary to cope with the changing environment should be initiated by the system rather than by users.

Existing applications and services are not designed or capable of managing unique characteristics of mobility. They provide highly contrasting quality of service, such as

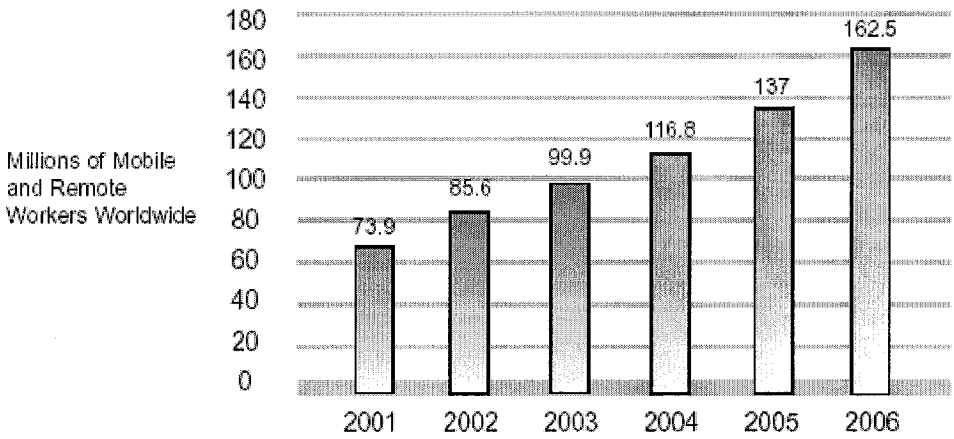

Figure 3. Number of Mobile and Remote Workers Worldwide

call set-up time and they do not adapt to the various devices capabilities. While roaming through wireless overlay networks, users are likely to experience rapid variations in the quality of communications and presentation. Such variations have an impact on mobile applications and supporting services.

Hence, it is crucial that mobile applications and supporting services can be developed which are capable of handling problems resulting from the use of wireless networks including the fluctuating quality of service, disconnection and various devices capabilities.

The challenges for mobile computing lie in three broad areas [6]:

- Providing reliable wireless communications services,
- Building applications that deal with the arbitrary disconnected nature of mobility, and
- Building applications that are not tied to fixed locations.

1.3. Thesis Proposal

1.3.1. General Aims

This thesis examines the issues and implications associated with supporting mobile users and their provisioning with personalized multimedia services wherever they travel and whichever devices they use. It has focused on the development of a framework to support the building of multiple mobile applications. The framework along with the policy-based approach makes it easier to build complex mobile applications.

More specifically, the key research aims of this thesis are:

- To examine the requirements for extending the agent model to support the development of mobile applications, capable of exhibiting appropriate behavior in an unreliable mobile environment.
- To control and manage the dynamic changes in locations, activities and profile of mobile users that enable authorized services to be fully personalized.
- To build and develop a set of generic policy-based agents those cooperate together to provide adaptive applications and services to mobile users. Agents make use of different network services and therefore off-load the need to deal with them directly by users or applications.
- To examine the implications for developing novel applications and services which make use of the cooperative policy-based agents to support mobility and context-awareness. Such applications include "virtual teams" collaborative work and ad-

hoc communication facilities for mobile users, and multimedia services provisioning for mobile users while moving from one site to another.

- To evaluate the effectiveness and adaptability of the cooperative agents. More specifically, this effectiveness should be judged on the ability to access local services at visited sites using the personalized preferences available at the home site and automatic authentication and configuration of authorized services to be performed at visited sites.

The management of a group of mobile users that could be highly distributed referred to as a *virtual team*, and their provisioning with collaborative multimedia services are of importance to this work. Hence the requirements for the virtual team management are described in more detail below.

1.3.2. Virtual Team Management

Virtual team issue has emerged as an important new teamwork model, distinguished from the conventional way in which people work by its ability to transcend distance, time and organizational boundaries. A virtual team consists of a dynamic collection of (possibly mobile) users, a set of collaborative services and network facilities that ensure a flexible and secure coordinated resource sharing.

We considered various requirements that could enable an efficient relationship among a recognizable team of mobile users, a set of collaborative services, a set of network capabilities and a collection of rules binding these elements together such they behave in a consistent manner to satisfy team needs. For an environment supporting the integration of such entities, we have identified the following specific requirements:

- Ability to create and manage a virtual team context that enables a full customization of services.
- Automatic authentication of team members, and configuration of collaborative sessions without requiring their direct intervention.

- Enabling a seamless physical resource allocation of logical resources requested when creating the team.
- A separation between team services (applications) and network services (e.g. mobility location, Class of Service (CoS) / Quality of Service (QoS), multi-party, peer to peer, multimedia session control), so that a team service may dynamically be adapted to network conditions and team's context.
- Distributed control and management of dynamic changes that may occur during the team life cycle, either in its structure, its activity and planning, or in its members' locations.

1.3.3. Applications Prototypes and Scenarios

Within the scope of this thesis, three different applications have been discussed and prototypes of these applications have been designed and implemented using policy-based agents that compose our framework. Our goal is to demonstrate that our framework architecture can support a number of different applications that make use of mobility and context-awareness.

The applications we will present are:

- *Multimedia Service Provisioning Application*: This application describes a set of cooperative agents distributed over different sites that work together to provide personalized services for mobile users over the Internet. Users moving outside the office (home site) are able to maintain an office-like environment at home, or at temporary locations (visited sites). Based on the concept of Virtual Home Environment (VHE), the Universal Mobile Telecommunication System (UMTS) proposes a service infrastructure that allows users to access their services from any suitable point; however this work is motivated by the desire to provide mobile users visiting another site with local services which reflect the pre-determined, personalized preferences available at their home sites. If equivalent local services are costly, or not available, a user would be able to access home services. At a visited site, a user profile is therefore determined based on an inter-agent

negotiation that accommodates agent autonomy and the privacy of sites information. This privacy could be achieved by introducing policies to manage and monitor the agents' behaviour and their decision-making.

- *The Management of Virtual Teams Application*: This application is designed to support multimedia and multiparty applications for geographically distributed teams across heterogeneous networks and devices. Much of the current work on collaborative systems tends to use the network only for transport, and attempts to single-handedly provide all the groupware necessary for virtual teams. The result is a very general set of capabilities to serve a broad user base. The motivation for managing virtual teams is based on the need to have the mobility support and team context as part of the environment middleware, which will enable its use across a diversity of content applications and collaborative services. As a result, a broader range and potential customization of these applications and services will be achieved and different mobile users will be provided by different levels of reliability and awareness.
- *The Ad-hoc Group Communication Application*: The application aims to provide dynamic real-time multimedia services to mobile users that come together in a physical meeting room with their own mobile devices, and start a collaborative session without any prior human configuration. Users are spontaneously connected with each other, and may share all the resources and services in the room. These resources and services are either provided by the room authority, or brought by users. Unlike virtual team scenarios where participants meetings are pre-scheduled, the ad-hoc group meeting scenarios are set up on the fly with users and services joining and leaving the meeting room in a dynamic fashion. Their presence is identified automatically by context awareness entities, and their devices are dynamically connected and supplied with appropriate tools.

1.4. Thesis Outline

The thesis is organized as follows. Chapter 2 reviews the related research for this work. This includes a discussion on existing approaches for supporting mobility in open distributed systems, personal mobility and group-based mobile applications. It also provides background information on mobile agents and policy management mechanism. Chapter 3 introduces the policy-based agent approach to support mobility. Chapter 4 presents a set of system agents' architecture that deals with the adopted approach. The designed and implemented cooperative system agents could support diverse applications and services for mobile users. Two particular applications that have been built upon these system agents are also introduced in this chapter. Chapter 5 and 6 discuss each of the applications prototypes that have been implemented using the framework. Chapter 7 evaluates the framework performance and validates our approach. Finally, in chapter 8, we present our conclusions and suggest future work.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Introduction

In order to provide mobile users with an office-like environment as they move from one location to another, their current location must be traced, the devices being used must be identified, and a cooperative relationship among home and visited sites must be established. The Universal Mobile Telecommunication System (UMTS) proposes a service infrastructure based on the concept of a Virtual Home Environment (VHE) [3]. The VHE seeks to present users with consistent personalized features, user interface capabilities and services in any network, at any terminal, and wherever the user may be located [3]. UMTS allows users to access their services from any suitable point; our work is motivated by the desire to provide mobile users visiting another site

with local services which reflect the pre-determined, personalized preferences available at their home sites. If equivalent local services were costly, or not available, a user would be able to access home services. At new locations, therefore, a user profile is determined based on an inter-agent negotiation. We propose a negotiation model that accommodates agent autonomy and the privacy of site information by introducing policies to manage and monitor the agents' behavior and decision-making.

Agents have been proposed in several research projects in mobile computing. The interest in agent technology to support mobility is mainly motivated by intrinsic properties such as its autonomy and its ability to migrate [8]. This allows a temporary disconnection between the end device and the network, while agents continue to perform their tasks. They return results when the mobile user reconnects [9][10][11]. Connection and disconnection permits an efficient use of connection time [10]. This is crucial, as mobile users often have to work in unstable network conditions [8], especially with wireless connections.

Although agents provide flexibility by migrating from one location to another while preserving their states, an efficient control and management of the system agents is needed. The use of policies is capable of achieving this control and enables dynamic changes in the behavior of the system agents in a flexible and scalable manner. Policy-based management has recently been developed by organizations such as IETF (Internet Engineering Task Force). IETF defines an information model for specifying a policy and also defines a framework for policy-based management and protocols for policy delivery and enforcement [12]. Policies are introduced in many research activities, but mainly focused either on general aspects of policies such as policy specification and conflict analysis, or on specific policy-based applications, such as network management [13][14]. We propose to apply policies through several levels of the system including security, admission control, user profile, resource management, and service personalization. We propose also to attach policies that are related to an agent's mission to that agent. By integrating policies with an agent, they become in effect part of the agent's model.

The policy distribution approach enables mobile users to be provided with personalized services. Each agent's policies could be translated and enforced according the context of its use, and be represented by a set of context policies. The policy translation and enforcement are therefore tightly bound to the context in which it is evaluated. Based on the agent's state, tasks or conditions of services and resources, a policy will be triggered.

Mobile agents will have therefore all the necessary control information (i.e. policies) to provide mobile users with personalized services at different locations. However, tracing mobile users' current locations and identifying the devices in use are needed.

2.2 Agent Paradigm

2.3.1. Agent Definition

There is little consensus among researchers about a common definition for an agent [15]. Pattie Maes [16] exhibits three crucial properties of agents: they are situated in a complex environment, autonomous, and goal-oriented. "*Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed*"[16].

IBM's [18] definition added the intelligence property to agents that employ some knowledge or representation of the user's goals or desires.

Agents are best described through the set of abilities they are characterized with. Figure 2.1 is a Bradshaw's figure [19] that classifies agents according to their abilities to learn, co-operate and act autonomously. Four types of agents are derived from these three minimal characteristics: collaborative agents, collaborative learning agents, interface agents, and smart agents.

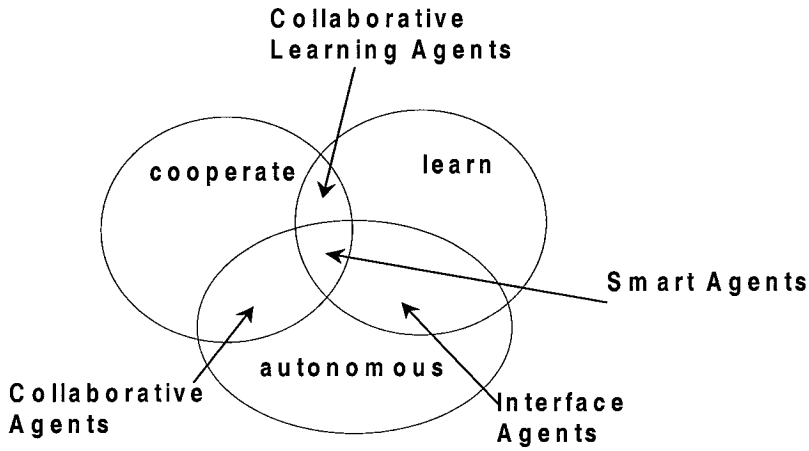

Figure 4. A view of an agent topology

Agents that have the ability to migrate from one host to another are called mobile agents. As it travels, the mobile agent performs work on behalf of the user, such as collecting information or delivering requests. This mobility greatly enhances the productivity of each computing element in the network and creates a uniquely powerful computing environment well suited to a number of applications and services.

In the following, we identify five main abilities that characterize an agent:

- *Autonomy*: Its ability to operate independently, without the intervention of human beings or other software entities, according to its internal state and its own set of goals and tasks that have been delegated to the agent.
- *Reactivity*: an agent must be capable of perceiving its environment, and responding to the changes that occur in it.
- *Pro-activity*: an agent should also be capable of taking some initiatives and performing sub-tasks that are required to satisfy a specific goal.
- *Sociability*: an agent should interact and co-operate with its peers and with end users in order to share or retrieve information, to exchange services, to provide end users with feedback, etc…
- *Mobility*: An agent should be able to migrate from one location to another to perform a specific task.

Several applications can benefit from the use of the agent technology [1]. In the domain of information discovery and retrieval, mobile agents can be assigned the goal of satisfying the search criteria of the end user.

In this case, the mobile agent will travel to the different servers on the network that may contain the information. The agent will filter the data of the server locally and only the documents that are relevant to the user will be sent back. This approach reduces the utilization of network resources and has proven to be an efficient approach for multimedia document retrieval [20].

Mobile agents can also be used to offer better support for mobile users. In fact, mobile devices generally have limited processing power and limited network connection time and bandwidth. A user can use his mobile device to create a mobile agent expressing the tasks the user wants to perform. The user will connect to the network to dispatch the mobile agent and disconnects again. The agent will travel to some dedicated servers in the network that contains all the resources and the knowledge bases required for the tasks of mobile agents. The agent will get reconstructed at the server and will start executing. The result, if any, will be stored on the server so that the user can retrieve it when he reconnects to the network. This approach reduces the connection time requirement of the mobile users, and allows them to exploit the processing power of the network servers. It has been used for supporting mobile users of database systems [22].

Mobile agents were also used in the field of network management for the distribution of processing and control [23]. The mobile agent travels to the different network devices and inspects them locally. The mobile agent thus gets access to more information and better control over the devices than a centralized network management system.

With the growing interest that agents are receiving as a new potential concept, a number of agents' platforms have been designed and implemented to provide the physical infrastructure in which agents can be deployed.

FIPA and OMG's MASIF are two important agent standardization efforts, which are attempting to support interoperability between agents on different types of agent platform [24].

Formed in 1996, the Foundation for Intelligent Physical Agents (FIPA) [25] is an international non-profit standardization organization that promotes the development of specifications of agent technologies. By following FIPA standards, a high level of interoperability within and across agent-based applications is achieved.

Mobile Agent System Interoperability Facility (MASIF) [26] is the first mobile agent standard of the Object Management Group (OMG) accepted in 1998.

While MASIF is primarily based on mobility of agents migrating from one platform to another via CORBA interfaces and does not address inter-agent communication [32], FIPA specifications has focused more on agent communication via ACL (Agent Communication Language) and did not address the issue of mobility at least in FIPA 97 standard (some aspects of mobility are provided in FIPA 2000 specifications).

There exist many platforms and toolkits that provide an agent execution environment with different functionalities and characteristics. Some of them are standards compliant (FIPA and/or OMG-MASIF) others do not support mobility or allow the code migration without the migration of the execution state of the agent. They may also differ about the communication mechanism (ACL, sockets, message-passing, RMI ...etc) and the security issue.

The list of agent platforms presented in the following sections includes FIPA-OS, JADE, Grasshopper, Zeus and Aglets. They are publicly available agent platform implementations and relatively popular among users and developers.

2.3.2. *FIPA-OS*

FIPA-OS [27] is an Open Source community project that enables rapid development of FIPA compliant agents. The core components of FIPA-OS are based on the FIPA reference model as shown in figure 2.2.

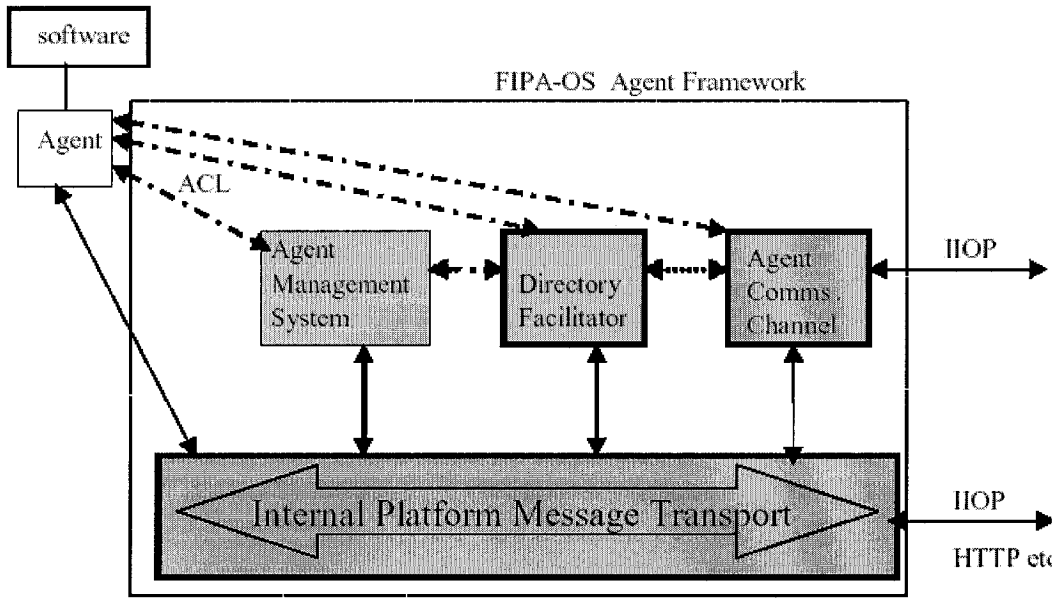

Figure 5. FIPA 97 Agent Reference Model

The Directory Facilitator (DF) provides "yellow pages" services to other agents. The Agent Management System (AMS) provides white-page services and life-cycle management services for agents and the Agent Communication Channel (ACC) supports inter-agent communication. The ACC supports interoperability both within and across different platforms. The Internal Platform Message Transport provides a message forwarding service for agents on a particular platform [27].

Publicly available under Emorphia Public License, FIPA-OS was (or is) being used in several projects and application domains including FACTS [28], CAMELEON [29], CASBAH [30], and CRUMPET [31].

2.3.3. Grasshopper

Grasshopper [32] is a Distributed Agent Environment (DAE). As shown in figure 2.3, the DAE is composed of regions, places, agencies and mobile/stationary agents. The region concept facilitates the management of the distributed components, i.e. Agencies, Places and agents. The agency is the actual runtime environment for agents and handles one or more places. Each place consists in a logical grouping of functionality of agents.

Grasshopper, which is developed by the IKV++ Technologies AG (IKV++ GmbH), is OMG's MASIF compliant, but has been extended by the 'FIPA Add On' package to support FIPA standards [32].

Grasshopper is used in several European projects such as CAMELEON, EURESCOM P815 and FACTS.

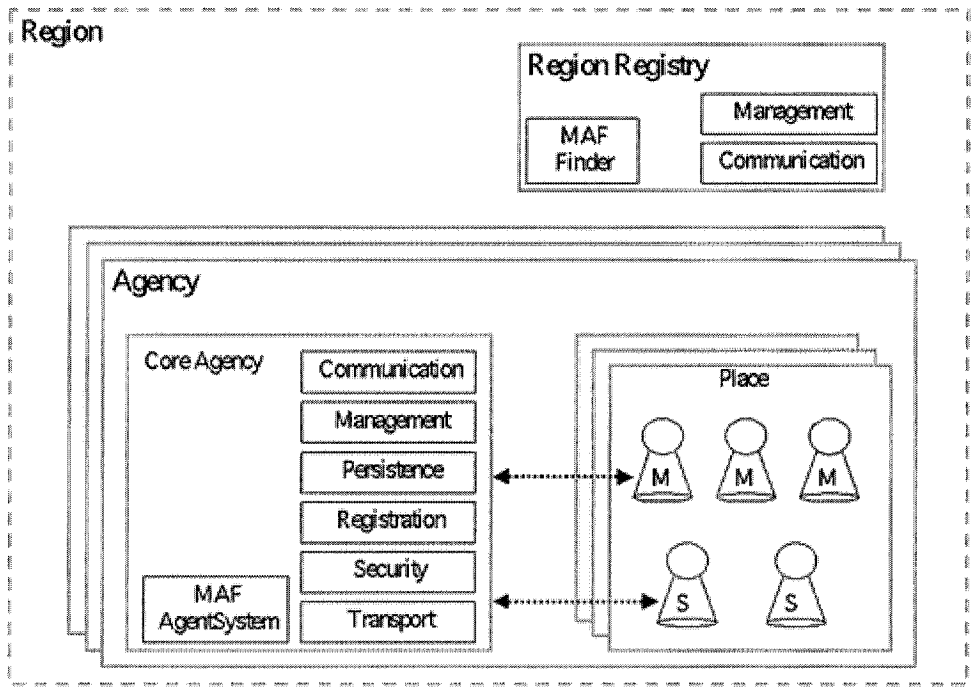

Figure 6. The Grasshopper Distributed Agent Environment

2.3.4. JADE

The Java Agent Development (JADE) is a software framework for developing agent-based applications in compliance with FIPA97 standards. As FIPA-OS, JADE [34] has implemented all the mandatory components of FIPA specifications, i.e. DF, AMS and ACC.

As shown in figure 2.4, JADE can be distributed over several hosts. Each host has one basic agent containers that provides a complete runtime environment for agent execution. Different agent containers launched on different hosts or on the same host are connected with one main container that represents the agent platform front-end. The main container distinguishes from the basic containers by supporting the DF, AMS and the ACC, where the basic containers support the DF only.

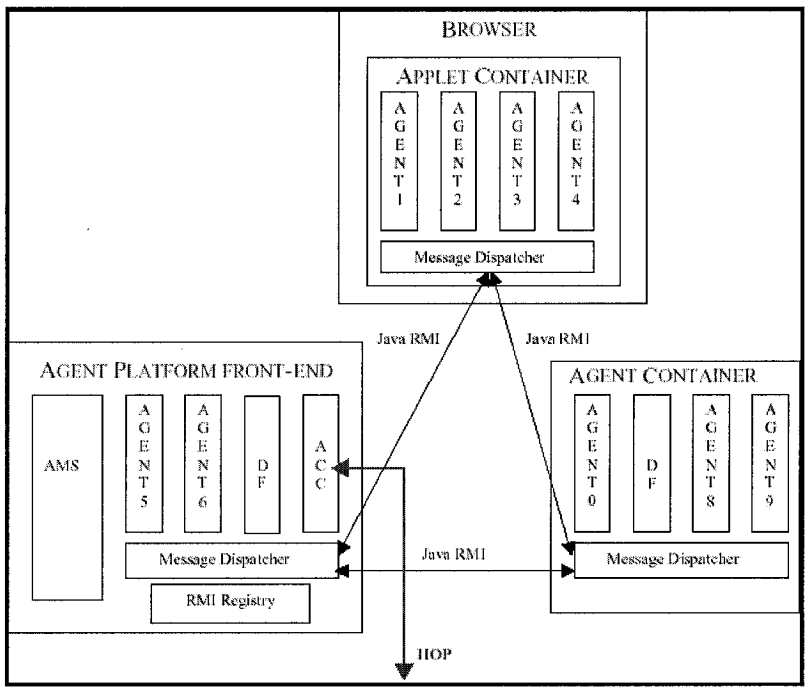

Figure 7. Software Architecture of one JADE Agent Platform

JADE is free software that is distributed by TILab. It is used by a number of universities, companies and projects such as FACTS, DICEMAN and FILIGRANE (e-commerce of mobile agents).

2.3.5. Other Platforms

A large number of other agent platforms exist such as Zeus, Aglets, Voyager, Concordia, Odyssey and AAP to name but few.

Zeus [35] provides a set of software components and tools, which are used to design, develop and organize agent systems. It provides high-level abstractions for developing intelligent goal-directed agents, but does not yet support the FIPA agent management specification [27]. **Aglet** [36] is an environment for programming mobile Internet agents in Java. As Java Applets, Aglets are Java objects that can move from one host to another on the Internet. The communication is done via the whiteboard mechanism and message-passing scheme that supports coupled asynchronous as well as synchronous peer-to-peer communication. **Voyager** [37] is also a Java-based platform from ObjectSpace that presents the particularity to be the most tightly integrated to the world of Java object-oriented programming. It is considered as an enhanced Object Request Broker (ORB).

The April Agent Platform (**AAP**) is also a FIPA-compliant solution for developing agent-based systems, but it is implemented using the April language, where most agent platforms are implemented using Java.

2.3.6. Lightweight Agent Platforms for Mobile Devices

Agent Platforms described in the previous sections are not suitable to pocket-sized mobile devices such as smart phones and PDAs. **MicroFIPA-OS** [38] and **JADE-LEAP** [39] are two lightweight frameworks for building agents targeted at mobile devices. They provide functionalities similar to FIPA-OS and JADE respectively, but with certain limitations due to the capabilities of mobile devices including the computing power, the amount of memory and storage space, connectivity and screen resolution.

Developed by the University of Helsinki within CRUMPET project, MicroFIPA-OS can be built as an independent platform or as a part of a FIPA-OS platform.

MicroFIPA-OS is Personal Java compatible that is supported in the J2SE. Agents running on FIPA-OS and/or MicroFIPA-OS interact using FIPA-compliant HTTP communication.

JADE-LEAP represents a special agent container that communicates with the main container agent platform within JADE.

MicroFIPA-OS and JADE-LEAP are quite similar FIPA-compliant agent platforms, but MicroFIPA-OS is targeted to more powerful devices (i.e. PDA devices), where smaller devices can support JADE-LEAP.

FIPA-OS and therefore **MicroFIPA-OS** have been used as agent platforms for supporting application prototypes implemented within this thesis. JADE would be more efficient since, in addition off being FIPA-compliant and has a lightweight version for mobile devices; it supports some aspects of mobility (i.e. code migration without the migration of the execution state of the agent). At the time we started this work, JADE was not available as an open source. However, since the two platforms are FIPA compliant, we expect to use one platform in one location and the second platform in another one in implementing our prototypes.

2.3.7. Agent Communication Language

Inter-agents communication is one of the most important aspects of multi-agents systems. An Agent Communication Language (ACL) provides agents of means of exchanging information and knowledge based on speech acts [40].

FIPA ACL [41], developed by the Foundation for Intelligent Physical Agents, is a high-level, message-oriented, communication language and protocol that allows agents to interact with each other, without dependence of transport mechanism (IIOP, HTTP, or another), of the content language (KIF, SQL or another) and without dependence of the ontology assumed by the content.

Based on speech act theory, an ACL message is a communicative act or an action to be performed after being sent. It consists of a set message types, the 'communicative

acts' or performatives, followed by the description of each act that are used to effect the mental attitude of the sender and receiver agents (their belief, desire or intention).

```
(inform
      :sender A
      :receiver B
      :content "weather (today, raining)"
      :language Prolog
      :ontology Weather)
```

Figure 8. An Example of ACL message

An ACL message from agent A informing agent B about the weather might be encoded as shown in figure 8.

In this message, the ACL performative is *inform*, the content is "today is raining" expressed in Prolog language, the ontology that determines a common vocabulary between agents A and B is Weather, so that the meaning of 'today' and 'raining' will be interpreted in the same way by the sender and the receiver. FIPA ACL acts are listed in the table 2 (next page).

Table 1 lists a set of reserved performatives that have been defined by the FIPA ACL specification.

Table 1. Standard FIPA ACL Performatives

accept-proposal	agree	cancel	cfp
confirm	disconfirm	failure	inform
inform-if	inform-ref	not-understood	propose
query-if	query-ref	refuse	reject-proposal
request	request-when	request-whenever	subscribe

Table 2. List of FIPA ACL acts

Parameter	Category of Parameters
performative	Type of communicative acts
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Description of Content
encoding	Description of Content
ontology	Description of Content
protocol	Control of conversation
conversation-id	Control of conversation
reply-with	Control of conversation
in-reply-to	Control of conversation
reply-by	Control of conversation

2.3 Policy Management

Policy-based management provides a mechanism to allocate different types of resources according to defined policies. Network resources, such as network bandwidth, quality of service, and security are well known examples of resources to be managed by policies. For instance, as the requirement for QoS increases with the use of Voice over IP (VoIP) and other real-time applications, the requirement for policy-based bandwidth allocation increases.

A policy-based management system allows an authority (the system adminstrator) to define rules and manage them in the policy system. Rules usually take the form of "*if* <*condition*>, *then* <*action*>." A condition may be related to a particular user or group of users, a specific time of day, a type of service, a specific network access point, or a combination of specific events. Events are triggered either by a time-period condition, a change in the state of a resource or as a result of an action. An action is a method invocation which is related to a specific entity.

The use of policies in management was first introduced by Sloman [45]. His work was the trigger for other research activities focusing on policies. Sloman's work introduced policies and showed the power of this concept particularly in the context of distributed systems. However the focus was put on general aspects of policies such as Policy Specification [46], Conflict Analysis [47], Policy Domains [48] and Hierarchies [49]. Other research groups have focused on the use policies for specific applications. Applications may vary from Network Management as described in [50] [13][14] to Collaborative Systems as described in [51].

The IETF Policy Framework (POLICY) Working Group has developed a policy management architecture that includes the following components:

- **Policy Management Service:** A graphical user interface for specifying, editing, and administering policy.
- **Dedicated Policy Repository:** A place to store and retrieve policy information, such as an LDAP server (Lightweight Directory Access Protocol).
- **PDP (Policy Decision Point):** A policy server that is responsible for handling events and making decisions based on those events and updating the PEP configuration appropriately.
- **PEP (Policy Enforcement Point):** PEP enforces the policies based on a set of rules that receives from the PDP. The PEP may be located in network nodes, users' devices, service provider ...etc.

A variety of protocols may be used to communicate policy information between the PDP and the PEP. COPS (Common Open Policy Service) is the usual protocol which is a client/server protocol that provides transport services for moving policy information among different points. COPS also provides the transport for policy queries and responses.

A typical IETF policy transaction starts with a request to access some entity (i.e. device, network access point, serivce ...etc) by a user or another entity. For example,

a user may request access to a service interface at a particular location. The service interface forwards the request to the PDP in the policy server using the COPS protocol. The policy server then queries the LDAP directory to determine the user's authorization. The information is then sent back to the policy enforcement point (PEP) which enforces the policy decision to the service interface.

Policy-based management is best for systems where the execution environement changes significantly during the life cycle, such as in mobile applications and network management. For static applications that do not require significant changes at run-time, the benefits of deploying a policy-based infrastructure may not justify the costs. Figure 9 shows the key opportunity space for policy-based systems.

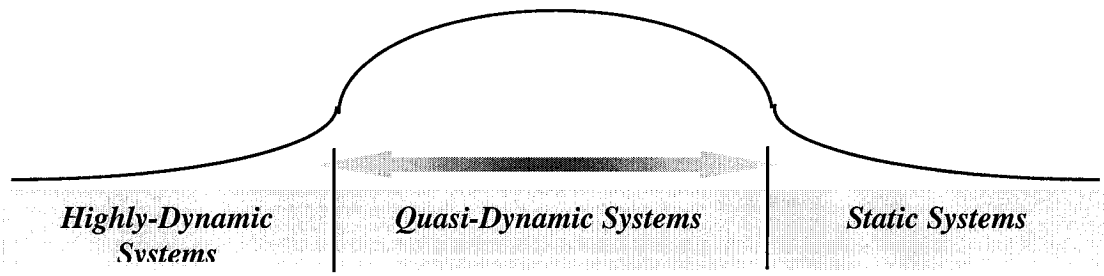

Figure 9. Key opportunity space for policy-based systems

In this work, we propose the Policy Service Agent (PSA), which is application dependent, and the Policy Management Agent (PMA), an application independent tool, for managing policies at a high level.

Using the PMA, the policy adminsitrator needs only to specify high-level policy rules to achieve application policies. These policies are captured by the PSA and then deployed on individual agents where high-level policy rules are translated into agent specific policies to be enforced in application agents.

Such a design approach has the advantage of allowing the applications initially designed with no policy abilities to become policy based. That could be achieved in two steps: (i) developing a PSA for this application and (ii) register the application agents with the PSA.

Another important aspect of this work is the use of policy-based agents. In previous work, researchers have used agents mainly to monitor and enforce policies [25]. A new initiative by FIPA attempts to use policies in the context of agent's platform but remain at a preliminary stage. Furthermore, this specification does not mention any effect of policies on the agent model. In contrast with this, we have considered in our work agents as being the basic entity in the application. An agent wraps every external software or hardware, and each agent manages its own set of policies. The agent refers to the PSA only for application-level and network-level policies.

2.4 Group-based Mobile Applications

Mobile applications are software applications that run on mobile computers, such as laptops, PDAs, palm-tops, and other handheld devices. An application is collaborative when it supports groups of people engaged in a common goal and provides a shared interface that allows users to collaborate with each other. A collaborative application (a) interacts with multiple users and (b) links these users to receive input from multiple users and creates output for multiple users.

Mobile and collaborative applications are contained within the larger area of distributed systems as shown in figure 10. Other related areas are multimedia and real-time computing.

Many collaborative environments providing a shared team workspace with common tool-set and shared applications have been developed. These systems generally attempt to single-handedly provide necessary services to serve a large broad user base, and tend to use the network only for transport. In contrast, our virtual team management service is strongly motivated by supporting mobility of users, providing them with different levels of reliability and awareness, and by enabling the integration of a variety of network services to satisfy the unique needs of diverse classes of virtual teams.

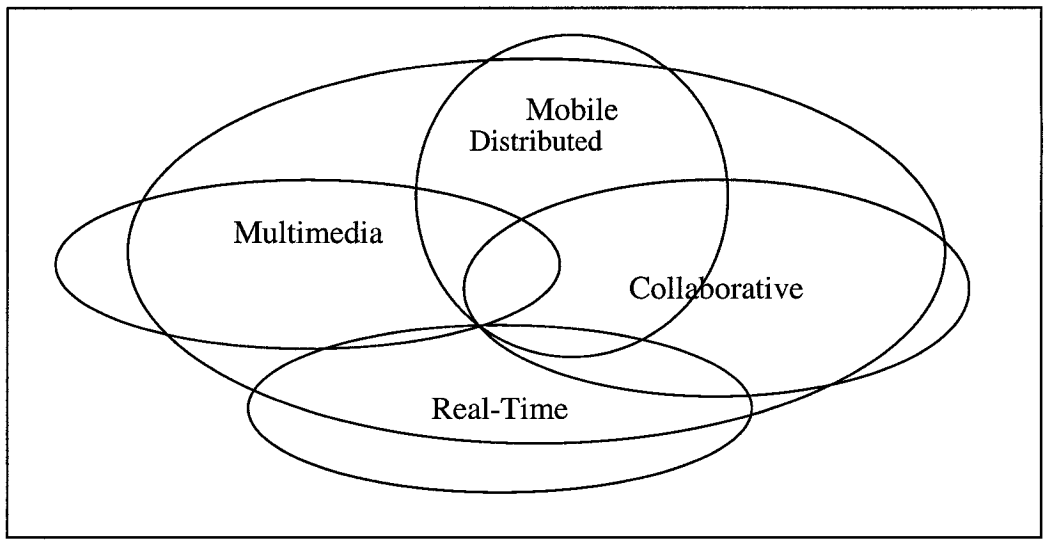

Figure 10. Areas of distributed mobile and collaborative systems

We also argue that collaboration in a static environment, where it is implicitly assumed that connectivity of all participants' devices is always-on, is different from collaboration in a mobile environment due to the weak and intermittent connectivity in such environments (i.e. a user may become temporarily unavailable while still engaged in a collaborative session). In a mobile environment, dynamic collaborative groups tend to be formed spontaneously and could be motivated by common interest and the situation of a group of mobile users, and not usually driven by a predefined goal.

The form of interaction among (mobile) participants may also differ from its traditional way defined in groupware systems. In a mobile environment, depending on her location and the used end device, a user may be more or less active in the session and her activity may be variable, for example, switching between synchronous/asynchronous depending on the user connectivity state.

Our approach has certain similarities to MoCA (Mobile Collaboration Architecture) [55]. Through basic services and via the ProxyFramework entity, MoCA exposes context information to the application developer such as the device characteristics and location, the quality of the connectivity and user preferences, which allows the

collaborative application to be more adaptable to network conditions and the overall environment.

YACO (Yet Another Collaboration Environment) [56] is also a framework for collaborative environments that use MobiKit which is a mobility service toolkit based on proxies [56]. A mobile user can inform collaborative users of his/her disconnection (i.e. moveOut operation), and when the user reconnects, all missed events are replayed by invoking the moveIn operation. YACO provides a messaging service, the service discovery, and a service for sharing various artifacts.

DACIA (Dynamic Adjustment of Component InterActions) [57] is a mobile component framework that allows groupware systems to adapt to available resources and to support user mobility. Using DACIA, components of groupware applications can be dynamically loaded, reconfigured and interconnected, and moved between hosts while maintaining connectivity with services and other users. It allows users to pull and drop application components from one host to another. The framework ensures that during reconfiguration or movement of components, the structure of the application and data flow is maintained.

The YCab [58] framework has a flexible API for the development of collaborative services for mobile users in ad-hoc networks. The architecture is divided into framework API and services. Services implement an interface providing basic methods required by the framework. The framework is responsible for sharing information between the services and other members in the collaborative session. YCab provides skeleton services that can be used to build custom services.

The aforementioned environments essentially try to shield from the application developer different aspects regarding mobility and user location, and aim to provide seamless and anywhere-available service. However, they do not use information about the current context for triggering appropriate adaptation of the collaborative application behavior or for enabling context-aware application policies.

The use of policies, combined with context-aware agents, enables the virtual team service to dynamically combine a collection of third party applications with a

selection of network capabilities, and off-loads the need of dealing with these capabilities directly within applications.

2.5 SIP and Mobility

The IMT-2000 universal mobility objective is a precondition for achieving multimedia session mobility. Universal mobility includes global connectivity and roaming, and access to the same services, with a consistent QoS, anytime, anywhere, using any terminal. Any technology that will succeed in supporting multimedia session mobility must also implicitly provide support for universal mobility, which includes the four mobility aspects: personal, terminal, service and session mobility.

Supporting multimedia session mobility over IP-based networks involves maintaining an active multimedia session on a mobile device, while the device is roaming across heterogeneous wireless networks.

Several approaches for mobile multimedia support over wireless networks have been investigated. They usually consider the network layer solutions such as Mobile-IP [101] and Cellular-IP [102] protocols. Recently, the Session Initiation Protocol (SIP) introduces an application layer approach for addressing mobility. This approach has the advantage to not make changes to the IP stack [2].

H.323 is also an alternative that provides multimedia session management at the application layer, but its lack of flexibility limits its potential and makes SIP more suitable and appropriate application layer solution.

2.5.1 Mobility aspects using SIP

SIP can be used to implement all four mobility aspects of IMT-2000: personal, terminal, service and session mobility. However, the terminal mobility would be more efficiently implemented using Mobile-IP.

SIP provides transparent support of name mapping and redirection services through the use of the SIP Registrar Server that keeps track of all the devices associated with a particular user with their associated addresses. This tracking ability provides SIP with

inherent personal mobility support – a user can use a single personal identifier all the time, regardless of the device(s) used or the network location. Incoming calls will be redirected to the appropriate active device. In case the user is associated with multiple devices, SIP can provide personal mobility support through the use of a SIP Registrar Server which determines dynamically the actual address of the active device when receiving a session initiation request.

Similar to personal mobility, terminal mobility can be provided in SIP through the use of the SIP Registrar and Redirect Server. As the terminal moves across (heterogeneous) networks new temporary identifiers (IP addresses) are assigned to the terminal. These are updated with the SIP Registrar by using the SIP REGISTER method. The current location of the device is always up-to-date so that messages can be redirected successfully.

Service Mobility is defined as the ability of the network to consistently provide personalized services to the user, with the expected QoS, regardless of the user's location. A possible SIP approach to QoS support is through appropriate resource allocation during the session initiation and hand-offs. To support access to authorized services, SIP uses a combination of AAA (Authentication, Authorization and Accounting) and SIP REGISTER methods.

SIP can also successfully implement session mobility through the use of the re-INVITE method. This is an INVITE method sent while a multimedia session is in progress. While maintaining the existing session alive, new terminal(s) can be added to the session and existing ones can be removed. Changes could be made to the parameters of the session in progress in order to match the capabilities of the newly added terminal(s).

Figure 11 shows a mechanism of performing session mobility through the use of a Third Party Call Controller (3pcc) [98].

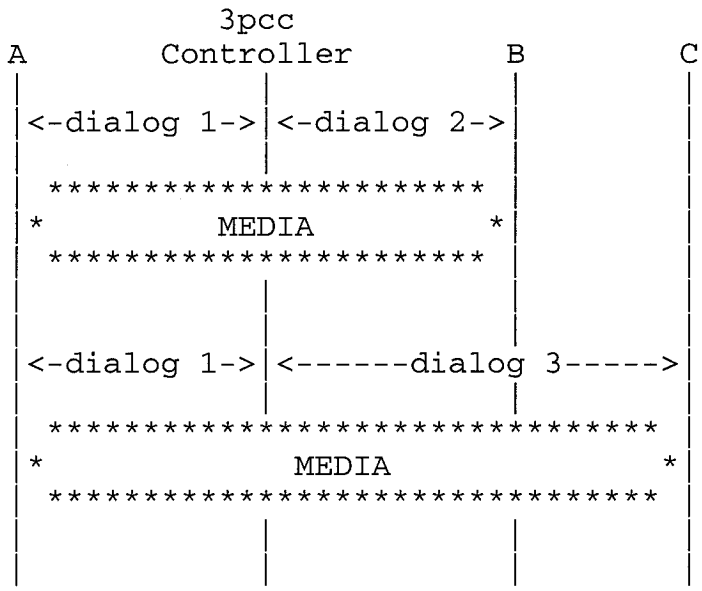

Figure 11: Session mobility using 3pcc

The 3pcc controller in Figure 11 has established a session between A and B using dialog 1 towards A and dialog 2 towards B. At that point, the controller wants A to have a session with C instead of B. To transfer A to C (configuration shown at the bottom of Figure 11), the controller sends an empty (no offer) re-INVITE to A. Since A does not know that the session will be moved, its offer in the 200 OK states that the current status of the media stream in the send direction is "Yes". After contacting C establishing dialog 3, the controller sends back an answer to A.

This answer contains a new destination for the media (C) and should have downgraded the current status of the media stream to "No", since there is no reservation of resources between A and C.

The call transfer could also be made using SIP REFER method. For instance, if A is in a call with B, and B decides to transfer the call to C, B's SIP user agent (UA) can send a SIP REFER request to A's UA providing C's SIP Contact information. A's UA will then attempt to call C using that contact. A's UA will then report whether it succeeded in reaching the contact to C's UA. REFER is a SIP method as defined by RFC 3515. The REFER method indicates that the recipient (identified by the Request-URI) should contact a third party using the contact information provided in

the request. The use of REFER method allows seamless session transfer from one user device to another.

2.5.2 SIP Scenario

A user may be associated with multiple network devices at the same time (i.e. phone, laptop, PDA, and desktop). SIP allows the use of a single logical address associated with multiple actual addresses.

For example, the logical address h_mmarl@genie.uottawa.ca may be associated with the email address harroud@site.uottawa.ca and phone number (613) 328-1234. When an INVITE message is received by the SIP Redirect Server for h_mmarl@genie.uottawa.ca all the addresses associated with this logical name are INVITEd to the session simultaneously. A session is established with the first device that answers the request, and the other INVITEs are cancelled. This disassociates the user from any specific terminal or network location.

2.6 Summary

This chapter surveyed the main concepts deployed in the thesis including the agent technology currently available to support the development of mobile applications, the policy management, and the use of session initiation protocol (SIP) in managing mobility.

Among different agents platforms, FIPAOS platform and its corresponding lightweight version for small devices (MicroFIPAOS) have been chosen for the implementation of our framework and the applications prototypes.

The use of policies at different levels enables agents to dynamically adapt their behavior depending on the situation of their use. Finally, SIP has features and characteristics which are capable of handling various forms of mobility, and therefore its integration to the framework is proposed.

CHAPTER 3

SIP AND POLICY-BASED AGENT APPROACH TO SUPPORT MOBILITY

This chapter describes the approach we have adopted for supporting mobility, so that mobile users could be provided with personalized services at different locations. The approach consists in designing and developing a framework that offers a set of services to support mobility by combining different concepts including the agent technology, the policy management, the context awareness and SIP presence. We introduced a new agent model by integrating policies that are related to an agent's mission to that agent. Each agent's policies could be translated and enforced according to the context of its use. The policy distribution approach enables mobile users to be provided with personalized services. Contextual information, which

includes any information that characterizes the user operating-environment, is automatically collected and then translated from different representation formats to a set of *context policies*. The policy translation and enforcement are therefore tightly bound to the context in which it is evaluated.

SIP, as explained in section 2.5 (chapter 2), provides mechanisms that permit determining the current location of users, devices and services, perform call setup using the session description protocol, and establish, maintain and terminate multimedia sessions. We propose the use of personal assistant as an entity that represents a user and may act on her/his behalf (personal secretary). The personal assistant will be instructed to REGISTER the user with SIP REGISTRAR each time the user moves to a new location, so that the current location of the user is always up-to-date. The personal assistant will also be capable of answering SIP calls of behalf of the user according to her/his instructions. The user may want to not be visible at certain locations or may want to filter the received calls according to a set of predefined user policies.

3.1. Policy-based Agent Model

Agents have the ability to act on behalf of other entities operating on their own without the need of human guidance. Hence agents have individual internal states and goals, and they act in such a manner as to meet these goals on behalf of users. To fulfill its tasks, an agent may migrate to remote sites and interacts with other entities at these sites. This means that the agent should be provided by some control schemes to govern its behavior and to make decisions autonomously. The controlling part of the agent could not be hard-coded into the agent's components at design time, because it does not allow applications to adapt to the evolving condition of a mobile scenario. The use of policies is capable of achieving this control and enables dynamic changes in the behavior of the system agents in a flexible and scalable manner.

A policy-based agent is an agent to whom we attach a set of policies to monitor its behavior and decision-making. The clear separation between the coding parts of the agent (i.e. tasks) and the monitoring policies permits to dynamically adapt the agent strategies and its overall behavior to the context of its use without impact on its components implementation.

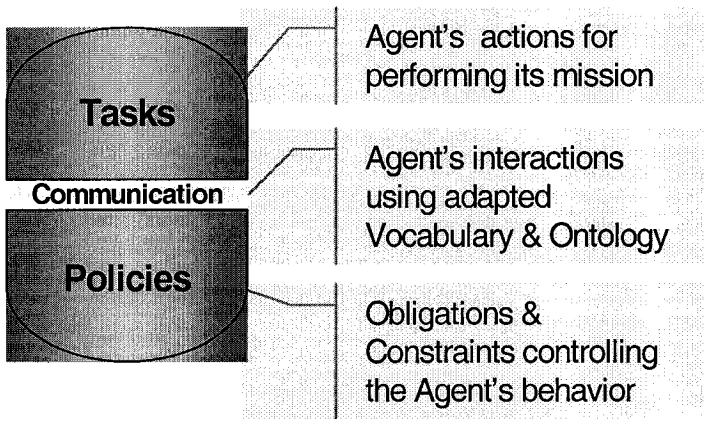

Figure 12. Policy-based agent model

Figure 12 shows the proposed policy-based model that is composed of three parts:

- **Tasks** that the agent is capable of performing. It includes the code and the agent's **state**.
- **Policies** that represent the dynamic part of the agent and allow the control and change of the agent behavior.
- **Communication** protocol, in order for agent to communicate and interact with its peers. Agents engage in high-level conversations using an ontology that determines a sort of common vocabulary that is shared between communicating entities.

The introduction of policies inside the agent provides it with more pro-activeness in facing new situations at visited sites, with roaming users and devices and with possibly discontinous interconnection.

Another advantage deriving from the policy-based approach is the increased

adaptability of the agent while changing from one context to antoher. While in a new context situation, an agent's policies could be translated and updated according to that context by interacting with the context-awareness entity that is responsible for collecting and interpreting the context information at the place the agent migrated to.

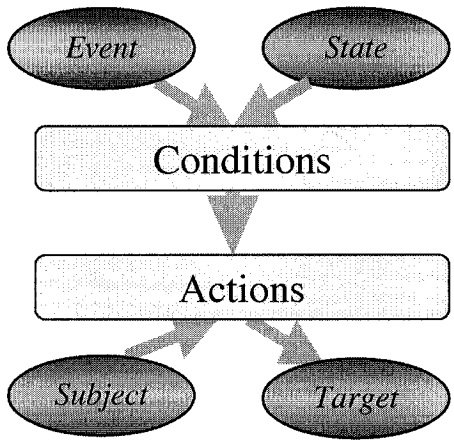

Figure 13. Definition of a policy

The context-awareness part is described in more details in this chapter (section 3.3).

3.2. Policy Distribution Protocol

Since policies and the agent code to which policies apply should be separated modules while moving together with the agent state, it is required to have a clear *policy model* that specifies how policies are expressed and represented and an *enforcement model* that determines how policies are evaluated and triggered, and hence, controlling the agent behavior.

3.2.1. Policy Representation

A policy is a set of actions to be performed by a subject agent on one or more target agents providing some conditions are satisfied or some events are triggered. A subject is the entity responsible for executing one or more actions of a policy. Targets are entities on which an action is performed. The conditions typically represent the state of one or more entities in the system (figure 13). Events are triggered either by a time-period condition, a change in the state of an agent or as a result of an action (i.e. before/after events).

A policy can be either an **obligation policy** that acts as the trigger for actions to be performed when conditions are applicable or an **authorization policy** that permits or prohibits actions on target entities.

Policies are applied to several levels of the system including security, admission control, user profile, and resource management. Examples of these policy levels are shown in tables 3 and 4.

Table 3. Sample of authorization policies at a visited site

Policy ID	Type	Subject	Action	Target	Conditions	Priority
P-1	pAP	Profile Agent	Service Access	User Assistant	[(cost>6, qos<v.s.); (cost>20)]	2
P-2	nAP	Service agent	Reservation	Resource Agent	[(space>10); (time>60)]	4
P-3	pAP	Service agent	Reservation	Resource Agent	[(cpu>100); (bandwidth>9]	3
P-4	pAP	Manager Agent	Encryption	Security Agent	[(key_len>64, site=MITEL); key_len>128, site=NRC)]	3
P-5	pAP	Interface Agent	Site Login	Site Context Agent	[(User.Site=NRC, User.dept=iit); (User.Site=Mitel)]	2
P-6	nAP	Interface Agent	Site Login	Site Context Agent	All	1

P-1 is a *positive* authorization policy (*permission*). It indicates that a particular visiting mobile user (represented by the user assistant) is authorized to use a particular service (specified in the service access action) if the local site assistant receives a proposal with a cost parameter value of more than 20 units. The cost may be less than 20 (and more than 6) but the quality of service will not include video streaming.

P-2 is a negative authorization policy (interdiction). It indicates that the resource agent is not allowed to reserve resources required by any service agent if the space requested is more than 10 units or processing time is more than 60 units. P-3,

however, grants lower-priority permission when CPU and bandwidth conditions are met.

P-4 allows the manager agent to request encryption action when communicating with the Mitel site using an encryption key length more than 64 bits. For the NRC site, the key length is more than 128 bits.

P-5 and P-6 are examples of admission control policies that specify the mobile users authorized to login at the visited site.

At run-time, a conflict between two or more policies may occur. The resolution of such conflict is based on the use of *Priority*: the one that has a higher priority will apply first. Conflicts may also occur when two or more agents communicate together. To resolve these kinds of conflicts, agents' negotiation takes place. The negotiation process is used to guide the interaction of agents as they make decisions on the user's behalf in accordance with their established policies. The negotiation protocol starts when an agent produces a proposal according to the local policies, and submits it to the other party. The other party evaluates the proposal, possibly generating a counter-proposal. This process continues until an agreement, or a negotiation deadline, has been reached. Proposals and counter-proposals consist of a set of elements (services or resources) associated with specific parameters such as cost, quality of service, or time.

Table 4. Sample of obligation policies

Policy ID	Type	Trigger	Subject	Action	Target	Conditions
P-7	OP	Resource reservation	Resource Agent	Notify ("critical resource mode")	Manager Agent	[(cpu<75); (bandwidth<8)]
P-8	OP	Resource released	Resource Agent	Notify ("normal resource mode")	Manager Agent	[(cpu>100), (bandwidth>10)]

P-7 and P-8 are obligation policies that apply to the resource agent. The resource agent has to notify a particular agent (e.g. the manager agent) if there is any change in the resource status (the trigger).

We represent policies using the Resource Description Framework (RDF) [60]. RDF is one way of representing metadata proposed by the W3C. It is used to represent the descriptive information understood by other RDF speaking communities. RDF follows the XML representation so that it is system independent and understandable. RDF represents a resource uniquely identified by a URI (Uniform Resource Identifier). Each of these resources or objects has a set of properties or attributes. A collection of these properties makes up the description of the object. The use of RDF representation was also motivated by the fact that FIPA-OS content language and agent profile parsers currently support XML\RDF encoding, which uses SiRPAC RDF parser.

```
<policy:Name rdf:about="Policy Name">Access Restriction</policy:Name>
<policy:PolicyRule rdf:about="Policy Rule">
 <policy:Enabled rdf:about="Policy Enabled">Yes</policy:Enabled>
 <policy:Trigger rdf:about="Policy Trigger">initialize session</policy:Trigger>
 <policy:Subject rdf:about="Policy Subject">TA</policy:Subject>
 <policy:Target rdf:about="Policy Target"> TSA </policy:Target>
 <policy:Condition rdf:about="Condition"> [media = video] AND
                                         [device=PDA;Phone]  </Condition>
 <policy:Priority rdf:about="Policy Priority">MEDIUM</policy:Priority>
 <policy:Action rdf:about="Policy Action">Reject</policy:Action>
 <policy:Audit rdf:about="Policy Audit">Yes</policy:Audit>
</policy:PolicyRule>
```

Figure 14. A sample of TA policy

Figure 14 shows an example of an agent policy in RDF format that specifies that a mobile user participating in a session via a PDA or a phone can only use audio, while other users may be allowed to use audio, video, or data conferencing. If the policy audit is enabled (i.e. Yes in the Policy Audit entry), the information about policy enforcement is stored, and could be analyzed by an authorized authority when needed.

RDF representation is also used to describe the capabilities of a device or a service together with the preferences of the user for content adaptation and negotiation using CC/PP (Composite Capabilities/Preference Profiles). CC/PP [61] can be thought of as a collection of RDF statements that describes the services and users. It is stored in the form of tables and each table is a collection of RDF statements.

The main purpose of CC/PP is negotiating content between two parties to facilitate the identification of a service. The RDF data in the CC/PP profile may originate from various sources and merged together into a single CC/PP profile. The description of the device may be from the device manufacturer and the preferences from the user's personal assistant. The content server compares this document with stored documents to find the best service for the user. Figure 15 shows a sample service profile that represents a printer.

```
<xml version="1.0" >
 <Document>
<Type value="Service" />
<Device>
      <Device_group value="PRINTERS" />
      <Device_name value="HP LaserJet 4050 N PS" />
 </Device>
<Properties>
      <Color value="no" />
      <sided value="yes" />
      <Memory>
           <minMemory value="8" />
           <maxMemory value="11" />
      </Memory>
      <Resolution value="1200" />
      <PS value="true" />
</Properties>
<Owner value="CBY-B502" />
<EnteringTime value="" />
<ExitTime value="" />
</Document>
```

Figure 15. A sample of Printer Profile

A mobile user profile similarly contains basic information such as Name, address, Home number, Designation, location etc. and some more parameters such as the status of the user, the services belonging to this user and security parameters of this user.

3.2.2. Policy Distribution

It is important to make a distinction between the process of defining, editing, storing and assigning policies, which is usually accomplished by an authority (i.e. administrator), and the process of interpreting and enforcing policies on specific agents at run-time.

The purpose of the first process, performed by the Policy Management Agent (PMA), is to assist the administrator of the system through policy editing. It is also responsible for detecting any static conflict between policies that may occur during the editing phase. These policies are considered high-level in terms of declarative rules. The rules are then translated to a set of low-level policies that specify which agent must perform specific actions, on which target agent, and under which conditions.

While the policy enforcement process is application dependent, since specific actions within the application are invoked and enforced, the process of editing policies could be used, in the same way, for different applications. Therefore, the PMA has been designed to be application-independent [62]. Information about the application is stored in the application profile within the Policy Information Base (PIB). The PIB is a sort of database that stores information about agents in the application, the services they provide, the resources they manage, and the policies that monitor their behavior.

Each agent-based application is associated with a specific PSA. All agents that make up the application have to register with the PSA, so that the application profile is recorded in the PIB. Policies are then edited through the PMA and submitted to the PSA. The PSA forwards those that are specific to each agent, and keeps those that monitor the overall behaviour. Events may be automatically intercepted by the PSA. Agents may also send events to the PSA for a policy evaluation.

Figure 16 shows the proposed policy distribution architecture for the management of policies from high-level policy rules to agent specific low-level policies.

Even though an agent triggers an action by evaluating its own policies, the action itself is not authorized if it is likely to use shared resources or to affect the state of the system. In that case, an authorization from the Policy Service Agent is necessary.

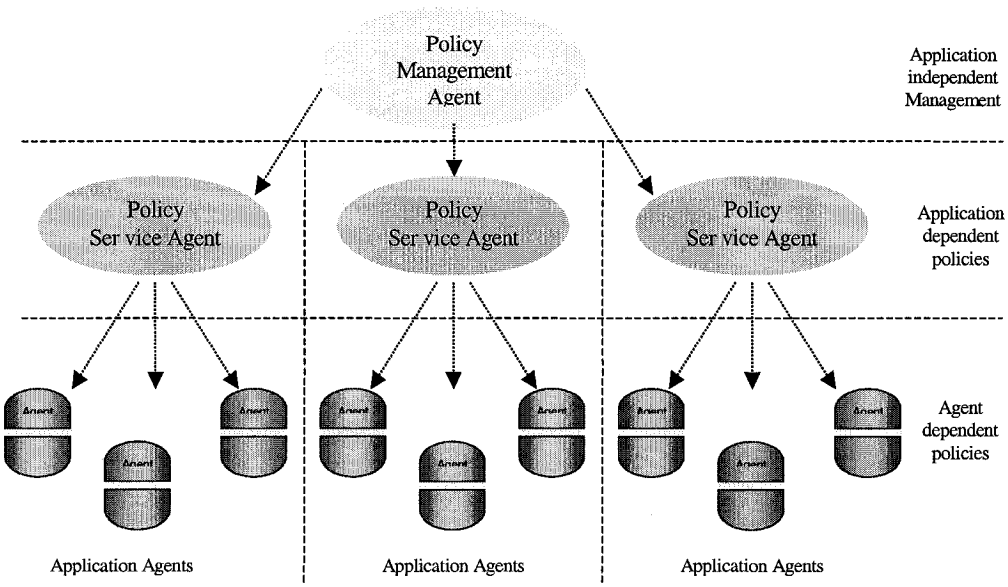

Figure 16. Policy Management Agent and Policy Service Agents

When an agent moves from one location to another it carries out its own policies. However, some of these policies may not apply at the new location, because the corresponding actions cannot be triggered according to conditions and the situation, referred to as *context*, prevailing at that location.

Collecting implicit contextual information through automated entities and making it available to agents at run-time let them to be context-aware and decide what information is relevant and how to deal with it. As described in the next section, collected context information are then translated to a set of context policies, so that agents behave in accordance with these policies.

3.3. Context-aware Agent Behavior

Managing context at the framework level clearly facilitates the development of mobile context-aware applications. This is achieved by providing the interaction mechanism with the entity responsible for discovering and gathering relevant context in the environment, so that appropriate information will be delivered to mobile users' applications and services. To successfully make use of context, we propose to integrate to the framework an entity that transforms the effective contextual information to a set of policies. Generated context policies are represented in the same manner as policies that are introduced manually by an authority.

3.3.1 Context Modeling

The term 'context-aware' was first introduced by Schilit and Theimer [63]. In their work, they referred to context as location, identities of nearby people and objects, and changes to those objects. Schilit et al. [64] claim that the important aspects of context are: where you are, whom you are with, and what resources are nearby. They define context to be the constantly changing execution environment, which is of three types:

- *Computing environment*: available processors, devices accessible for user input and display, network capacity, connectivity, and costs of computing
- *User environment*: location, collection of nearby people, and social situation
- *Physical environment*: lighting and noise level ...etc.

Contextual information may include any information that characterizes the user operating-environment, but in practice, four types are commonly introduced: **location**, **identity**, **time** and **activity**. The type location includes people and entities that are in or nearby (*where* the user is). The user's identity determines his profile, his role, and the relationship with other users ...etc (*who* is the user). Activities that are occurring (*what*) around or may occur at a certain time of day (*when*) determine the availability of resources (e.g. bandwidth and device capabilities) and connectivity to the network to name but few.

An example of contextual information use would be that when a mobile user at a certain location joins a collaborative session on the phone, he would use audio only and other materiel of the ongoing session (e.g. shared document) would be e-mailed to him or shown on a nearby terminal (e.g. fax machine) or even converted from text to speech depending on his current location and the availability of resources and services.

To allow a common understanding of the meaning of context information, ontology has been used as a paradigm for modeling and representing the context. The context is modeled in the form of *levels of expressiveness*, as shown in figure 17a, where the highest level is an abstraction of all concepts of context. As we go down through the levels, context is expressed in more detail and refined more concretely [65].

The highest level of abstraction is the *ContextView*, which provides a point of reference for declaring context information. *ContextView* represents different types of context that belong to a given entity. An entity could be a user, a service or the environment itself. There will be, at least, one instance of the *ContextView* for each entity in the environment. The properties *contains*, and *invokes* are properties of the *ContextView*. The classes *ContextFeatures* and *ContextEngagements* are the respective ranges of those properties. These classes are considered to be the second level of expressiveness in the ontology. Each instance of the *Contextview* will contain a subclass of the *ContextFeatures* that is *associated* with one or more *ContextDependencies* classes and is *invoked* by a *ContextEngagements* class.

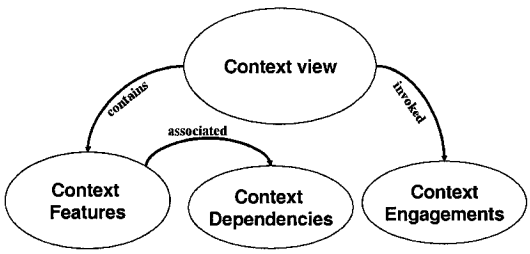

Figure 17a. the upper context ontology

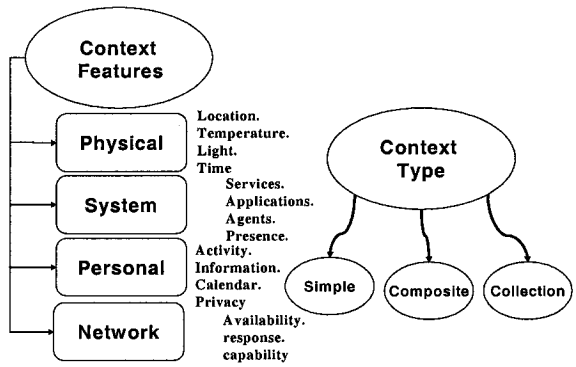

Figure 17b The defined context features and the associated context types

The *ContextFeatures*, shown in figure 17b, consists of the classes related to the physical environment (such as location), classes related to the system (such as services), classes related to persons (such as ongoing activities and identity) and classes related to the underlying network (such as availability and bandwidth.) Each of these classes is categorized to be either simple, a composite of other context information, or a collection of similar context information. More information about the developed context ontology modeling is given in [96].

The context acquisition, modeling and the context sensitive communication protocol are the main topics that are developed in our laboratory within a PhD thesis. More information could be found in [96].

3.3.2 Policy Generation

Managing the context at the framework level lets multiple domains and applications exchange contextual information and form a large-scale pervasive environment. However, generating context policies involves analyzing the context environment to determine which contextual information is meaningful for services that are provided to mobile users and do filtering from the global captured context.

The framework uses the Context Level Negotiation Protocol (CLNP). The CLNP [96] is a multi-attribute based negotiation protocol that allows automated context identification and agreements depending on each application and/or users needs. It permits the policy generator to subscribe and request contextual information and services implicitly, and to decide the way in which the requested information will be delivered and monitored. The protocol allows an application/user to send a context negotiation request to the policy generator, which in return maps the request to the context specification parameters and issues a negotiation request to the local context provider (i.e. the CMA).

The context policies that are generated by our framework could be applied in a variety of situations and at different levels.

3.4 Agent Negotiation and Policy Enforcement

Since agents have no control over one another, they must reconcile user's preferences and constraints depending on the visited site. Figure 18 shows the conversation graph used to monitor the negotiation process to achieve interaction between agents.

The negotiation starts by an agent A producing a proposal according to his own policies. The proposal is then sent to agent B that processes the proposal (Process state) and possibly generates counter-proposal or asks for additional information before making a decision (Asking state). This process continues until an agreement, or a negotiation deadline, has been reached (End state). Proposals and counter-proposals consist of a set of elements (services or resources) associated with specific parameters such as cost, quality of service, or time. Interactions among agents are

performed in exchanging information messages using the Agent communication Language (ACL).

In a proposal, values of different parameters are given explicitly by the sending agent, but in the counter- proposal, they are not quantified, in order to protect site privacy. Instead, they are given the values LOW, HIGH, UNEQUAL and VALUE. LOW indicates that the parameter needs to be increased in order for the proposal to be acceptable. An example would be when the proposed cost is insufficient for the request. HIGH shows that the parameter has to be decreased, such as for quality of service requirements. UNEQUAL indicates that the value of the parameter must be modified, such as when a password is not recognized. VALUE indicates that the parameter was not specified in the proposal and it should be assigned a value in next proposals. The sender agent uses its own strategy to assign new values to the parameters received in the counter-proposal.

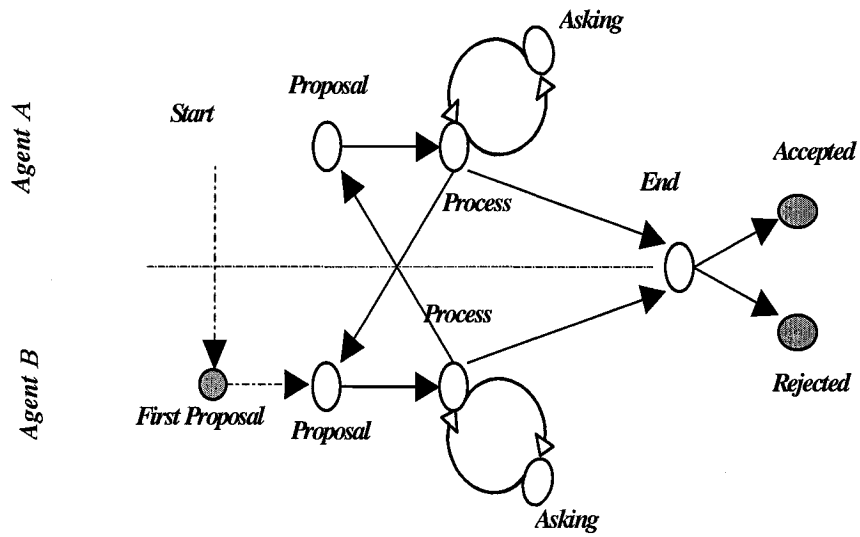

Figure 18. Agents Negotiation Graph

The negotiation continues until an agreement is reached based on the utility function calculation, which has the following characteristics:

- A lower bound on x (the normalized negotiated parameter). Satisfaction factor is considered as very small and therefore negligible below this value.

- An upper bound on x. Gain factor is considered as very small and therefore negligible above this value.
- Between these two bounds, satisfaction factor is monotonically increasing function of x.
- A controlling parameter "p" for projecting the user satisfaction.

The logarithmic function matches these requirements. The utility function can therefore be represented as a logarithmic function in the form of

$$U(x) = a.\ln(b.x + c) \quad (1)$$

$$a = \frac{1}{p - Ao} \quad (2)$$

$$b = \frac{e^{1/a} - 1}{R - A} \quad (3)$$

$$c = \frac{R - Ae^{1/a}}{R - A} \quad (4)$$

Where U(x) = utility function.

x = parameter value in negotiation.

p= user sensitivity parameter.

A_o= expected parameter sensitivity.

A= minimum parameter value.

R= maximum parameter value.

The utility function depends on the sensitivity parameter p. For every value of p, agents project their interest in a proposal specification according to the utility function shape. For example, for p values smaller than A_o, the user is more sensitive to larger values of the proposed parameter than to smaller values, such as quality of service parameter in service provisioning. For p values larger than A_o the user is more sensitive to smaller values than to higher values, such as cost parameter in service provisioning application.

Each agent decides the working utility curve according to the negotiation process. The optimal utility curve is accomplished by applying the following decision equations on the initial parameter value A_L.

$$U(A_L)_{x=A_L} = a.\ln(b.A_L + c)$$

$$If \quad U(A_L) \ll U(R) \quad for \quad A_L < R, \& \quad A_L > \left[\frac{A+R}{2}\right] \qquad 5$$

then $P < A_o$, agent is more sensitive to large parameter values and P is chosen from equation 2 to satisfy equation 5

$$If \quad U(A_L) \gg U(A) \quad for \quad A_L > A, \& \quad A_L < \left[\frac{A+R}{2}\right]$$

then $P > A_o$, agent is more sensitive to small context values and P is chosen from equation 2 to satisfy equation 5

After deciding the utility function curve for each parameter, an agent assigns a weighted priority value for each proposal that indicates its importance (as perceived by the user) in the overall negotiation. Agents use the priority values to differentiate between levels of proposals that would be proposed or counter-proposed.

The weighted priority value takes the form

$W_i = \frac{A_{Li} - A}{R - A}$ $\quad i = 1...N$ N= total number of proposal parameters.

The total satisfaction value will be equal to $\sum_i U(x_i).W_i$. The agent then decides if an agreement has been reached using this value, or if the negotiation should continue.

Agents' negotiation can be performed in the following situations:

- To determine a user's profile and preferences at the visited site. The negotiation process is performed between the home site assistant and the visited site assistant.
- To allocate resources required by the services requested the user at the visited site. The negotiation process is performed between the resource manager and the service agent that represents each service.
- To resolve any inconsistency or conflict that may occur during the evaluation and the enforcement of certain actions by the agents.

- To setup a meeting among a group of users, Personal Assistants that represent different participants may engage into a negotiation process to determine an appropriate schedule (time, date, duration and frequency) under the control of the site assistant, which plays the role of a mediator.
- To determine the contextual information that needs to be provided to a particular service at a particular location. The negotiation process in performed between the policy generator and the context manager agent, an entity that establishes the relationship with the context management system.

3.5 The Personal Assistant

FIPA defines a personal assistant (PA) as "*software agent that acts semi-autonomously for and on behalf of a user, modeling the interest of the user and providing services to the user of other people or PAs as and when required It is unobtrusive but ready when needed and rich in knowledge about the users and their areas of work*" [27].

A PA aims at providing intelligent assistance to the user in multiple ways depending on the user personal preferences. The PA assistance includes managing the user agenda, filtering and sorting e-mails, managing a user desktop environment, managing the user activities, locating and delivering (multimedia) information, purchasing desired items, and planning travels.

The PA is not restricted to provide assistance to any specific application for mobile users, but intended to be used in various applications. Examples of tasks that could be performed by personal assistants are as follows:

- Access and store most of the frequently used information; quickly search and locate documents of interest and importance.
- Organize information hierarchy according to users' personal settings and preferences.

- Reduce mobile users discomfort in using the touch-screen keypad and small-sized buttons by including visual figures and icons in its GUI.

- Assist user-oriented services such as e-mail retrieval, file/ media transfer, weather and other news reports gathering.

- Respond automatically to various requests that normally involve human users' intervention. It acts like a virtual secretary for mobile users. It is customizable and built for a single nomadic user to act and perform autonomous actions only on that particular user's behalf. Each user has his/her PA, just like their own backpack or agenda i.e., PA's are customized based on user's personal preferences and instructions.

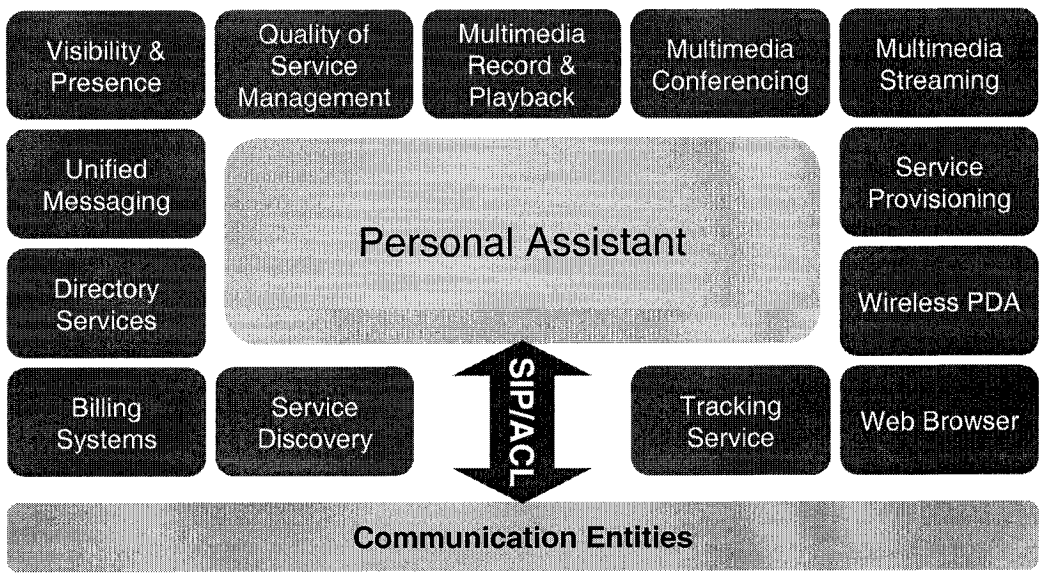

Figure 19. Sample of Personal Assistant tasks

Figure 19 shows an example of tasks where a personal assistant can help the user it represents to accomplish his routine jobs in different areas and domains. Different technologies and concepts can be used at both the design and implementation level. The session initiation protocol and the agent communication language are two communication protocols that can be used by the personal assistant components to communicate with external applications and services where the PA is involved.

FIPA specification describes a scenario of using personal assistants in arranging meetings among several participants located across companies with different calendar management systems. The scenario lends itself well to agent technology, due to the need for user profiling, integration of heterogeneous software, action on a user behalf, and local control, especially for the user calendar [27].

FIPA specification introduces also a User Interface Agent that constitutes a sort of translator between the human user and the personal assistant [27]. The interface agent is responsible for interacting with the human user and for making the other agents transparent to him. Moreover it is able to understand the user's requests and translate them for other agents. The nature of the personal assistant demands much more interactions with the human user than other type of agents. Consequently, enhanced user interfaces are required; especially those that are based on visual languages.

3.6 Summary

This chapter described our approach in managing mobility and especially personal mobility. Our approach combines four main concepts: (i) agent technology which is naturally chosen because of intrinsic properties of agents such as autonomy, migration and pro-activeness; (ii) policy management that allows agents to adapt their behavior dynamically; (iii) the context awareness mechanism as users move from one location to another; (iv) and finally SIP presence that permits to locate users and the devices that are used at anytime.

We introduced a new agent model by integrating policies that are related to an agent's mission to that agent. We also proposed to translate contextual information to a set of policies and defined a policy distribution approach that enables mobile users to be provided with personalized services.

To allow more autonomy and seamless users' mobility, we introduced an entity that represents each user and is capable of acting on her/his behalf: the personal assistant (PA). The PA is a particular agent that will be instructed by the "owner" user using policies and therefore behaves on his behalf according to these policies.

CHAPTER 4

SYSTEM AGENTS FRAMEWORK TO SUPPORT MOBILE APPLICATIONS

4.1. Introduction

This chapter presents the conceptual framework that provides necessary features and services to make the building of mobile applications easier. The framework allows developers to expend less effort on the details that are common across various agent-based applications for mobile users and focus on the specific objective of these applications.

The required features for the framework architecture are derived from the need to develop a flexible framework that adapts better to user mobility and available resources. As introduced in previous sections, these features are of several dimensions including the intermittent connectivity, devices capabilities, networking heterogeneity, services and applications dynamic reconfiguration, and users locations.

By addressing these features in the framework, we enable application designers to more easily build complex mobility and context aware applications. Designers will be able to leverage off of the framework, rather than be forced to provide their own general support in an individual manner. The framework also provides some basic services, so that it could be used without the need to add new components, but simply uses the existing support and built in agents.

4.2. The Agent-based Framework Architecture

We designed the framework architecture based on software agents. The logical architecture is shown in Figure 20. Due to the autonomy property of agents, there is no hierarchy among different agents composing the framework. Each agent interacts with its peers to fulfill its tasks. However, we presented the logical architecture of the framework in a hierarchical model depending on their tasks, so that we show at lower level agents that act at the network level up to agents that act at higher level (i.e. service agents).

The Network Service Agent (NSA) provides a simple interface to network services, and exposes them to users and group of users in a way that makes them easy to access and use. Network services include the ability to track the locations of mobile users, the quality and privacy of service, transport classes-of-service, and the multimedia session control protocols for the management of service provisioning or collaborative sessions.

A global data management level provides the functionality required to maintain the basics of user-related information such as users' profiles, their privileges and roles, a description of available services and resources, preferred quality of service, and policies repository. This level also provides the relationship that may exist between a

group of users, such as the list of the group members, the (possibly) shared resources and services lists.

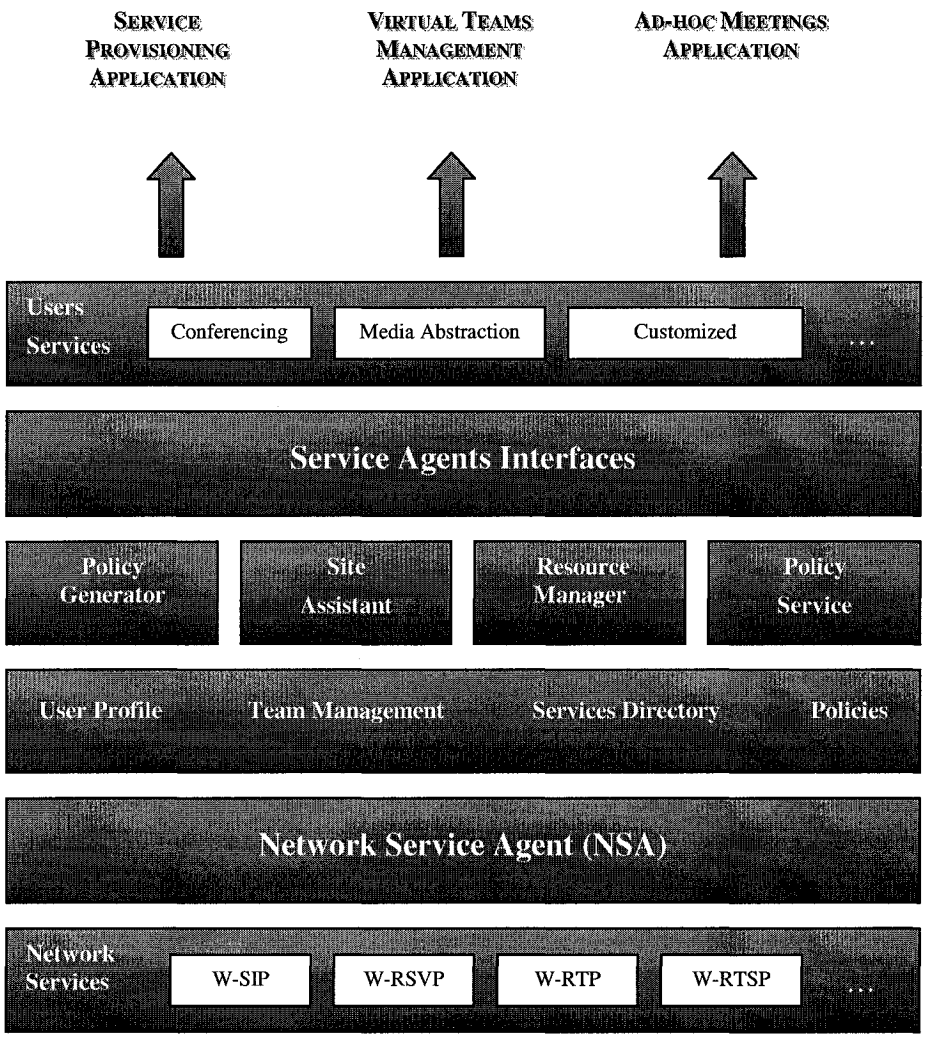

Figure 20. The Agent-based Framework Architecture

A set of system agents that support various mobile applications are then provided. The Policy Generator implements and interprets context information. The Policy Service Agent manages the policies at a specific site to control the behavior and decision-making of system agents. The PSA is responsible for locating events and conditions likely to trigger a policy. To enforce a particular policy, the PSA invokes a set of actions to be executed. For instance, the PSA may prohibit an authorization to a participant seeking to obtain a particular service or resource at a specific site.

The Site Assistant is in charge of preparing and setting up a temporary working environment to a user at the visited site. At the setup phase, the site assistant at the visited site establishes a negotiation process with the site assistant at the home site to determine the visiting user profile and the authorized services. A site represents a location that a mobile user is visiting such as a hotel, or a business company. It may also represent a room within a certain location such as B502 at the University of Ottawa, a meeting room or an office within an organization.

The Resource Manager is responsible for configuring the execution environment of each service at the visited site. This includes memory usage, processing time and network bandwidth, as well as the monitoring of resource consumption. A negotiation between the resource manager and the agent that represents a service must take place to make sure that the resources are available at the time the service is requested by the user.

Services such as the audio/video conferencing and the media abstraction service are offered to mobile users through Service Agent (SA) interfaces. Each service is associated with a SA that provides the necessary linkage and the execution environment. A wide range of services can be provided to mobile users by defining their corresponding SA.

4.2.1 The Network Service Agent (NSA)

The Network Service Agent (NSA) exposes a unique interface to allow the access to network resources. This agent makes use of different network services including the ability to track the current location of a mobile user, the quality and privacy of service, the classes-of-service transport, and the multimedia session control protocols for the management of service provisioning or collaborative sessions.

The NSA removes the need for users or applications to deal with network services directly. As shown in figure 21, the network is therefore not used only for transport; it also provides all the services necessary for mobile users to be fully customized. It relieves users of the constant need to be awa re of the details of their computing environment, thereby allowing them to focus on the real tasks at hand.

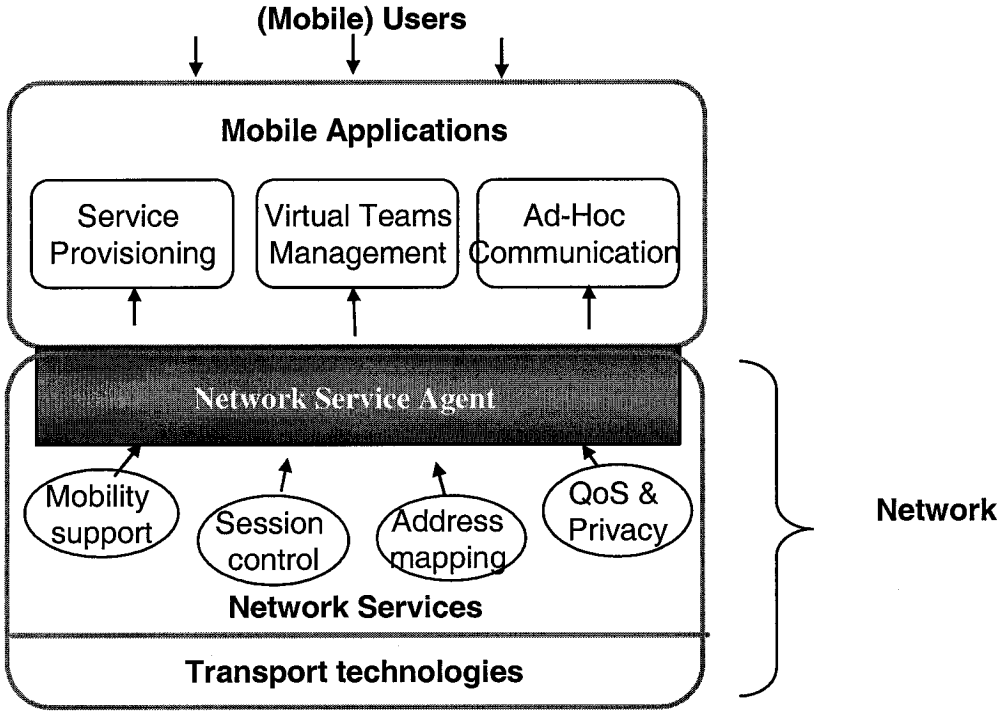

Figure 21. Network Service Agent Interface

With the NSA the complexity of different underlying networks, such as Bluetooth, 802.11x or WAN, is hidden and user services will use a uniform communication mechanism that is not dependent on the technology used to access the network (figure 22).

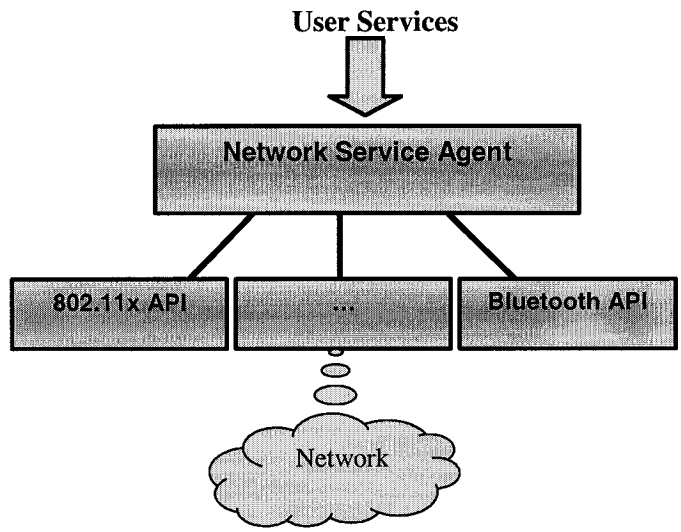

Figure 22. Network Service Agent and Underlying Networks

An example of network services that has been integrated to the NSA is the mobility support service that uses the Session Initiation Protocol (SIP) Proxy and SIP User Agent (SIP-UA) [66]. The description of this network service and its integration within the NSA are introduced in the next sub-section.

4.2.2 SIP Integration with the NSA

The Session Initiation Protocol (SIP) is an IETF signaling protocol that defines how to establish, maintain and terminate collaborative multimedia sessions. It provides mechanisms so that SIP User Agents and SIP Proxy/Redirect servers can determine the current location of a particular user and perform call setup including SDP (Session Description Protocol) for media description. When a user moves from one location to another, he sends via the SIP-UA, a REGISTER request to the SIP Proxy with his new location and/or his expected destination. At the time a call is performed (i.e. INVITE) the server determines the current location of the callee so that a connection can be established with the caller. We have developed a wrapper agent (W-SIP) that interfaces SIP-UA to the NSA for automatic mobile users' availability and presence. Figure 23 shows the SIP operation in the Redirect and Proxy modes.

There are several reasons for integrating SIP into the NSA:

- **Users Mobility**: Each mobile user is represented by a unique SIP address that is used to locate her/him by checking destinations she/he previously registered to. The SIP registration to the SIP server can be done on behalf of the user by the Personal Assistant as discussed in this section.
- **Session Negotiation**: The use of SIP allows a negotiation of the media type for establishing a session that makes use of one or more media streams (i.e. text, audio and video). The media description is performed by means of the Session Description Protocol (SDP). This information has been used to determine the capabilities of the user device.
- **SIP Presence**: SIP is particularly suitable as a presence protocol that conveys the ability and willingness of a user to communicate across a set of devices [67]. A SIP-UA, capable of receiving SUBSCRIBE requests, responds to them

by sending a NOTIFY message of changes in the presence state.

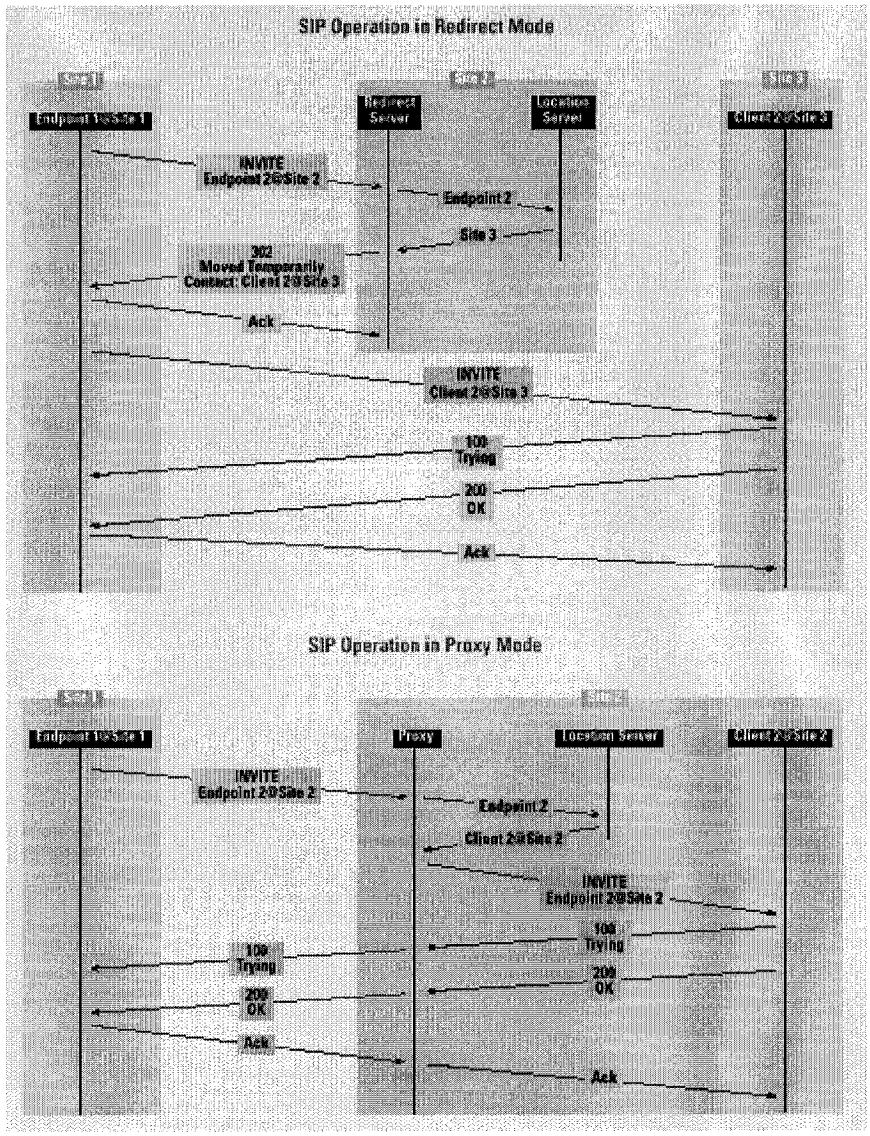

Figure 23. SIP Operation in Redirect/Proxy Modes [67]

To integrate the Session Initiation Protocol into the framework, a wrapper agent was created. The wrapper agent, W-SIP, establishes a connection between SIP-UA, a non agent-based component, and the agent-based framework environment, especially with the NSA as shown in figure 24.

At the time of setting up a session, the Site Assistant communicates with W-SIP to invite the user. When the SIP proxy locates a user, the SIP-UA sends a request to the personal assistant PA that represents that user, and waits for a response. Depending on the PA policies, the response could be either ACCEPT (to accept the call invitation), REJECT, BUSY or FORWARD. FORWARD specifies that no policy is triggered on behalf of the user. The SIP-UA then interacts directly with the user via a user interface, so the user can decide to either accept or reject the call.

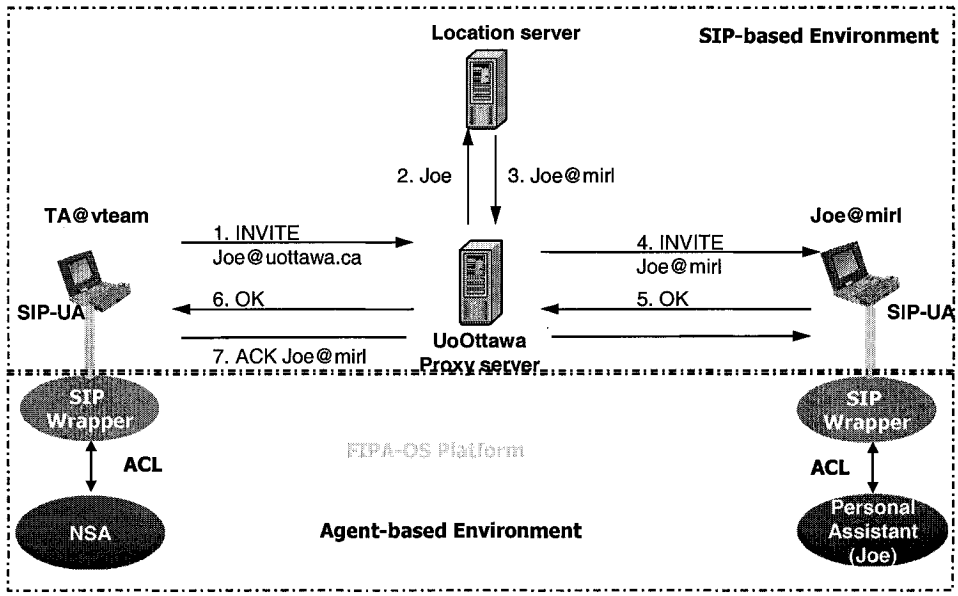

Figure 24. SIP Integration to the Framework

The response (ACCEPT, REJECT or BUSY from the PA or the user) is then sent back to the W-SIP with the agreed SDP description. These capabilities are then transmitted to the Site Assistant that uses them as conditions of its context policies to trigger appropriate actions to adapt services output to the end user device characteristics or to use a third party service for that purpose.

For instance, when a user starts using a small device which could be limited in its display resolution and bandwidth, he will likely be receiving a multimedia video stream service at a low resolution and frame rate, while on a large-screen display his video output will be at a higher native resolution of the display and at a higher frame rate. The suitable image formats and corresponding MPIs (Minimum Picture Interval, related to the frame rate) could be specified in the SDP description following the

media line, in order of preference. For example, the following lines in SDP would indicate that a device supports image sizes of 16CIF, 4CIF, CIF and QCIF (with the MPI for each format following the "="):

```
m=video 60300 RTP/AVP 34
a=fmtp:34 16CIF=8;4CIF=6;CIF=4;QCIF=3
```

In addition, the integration of SIP to the NSA allows the framework to control incoming calls. In deed, when a mobile user is hosted by a visited site such as a meeting room location, the site assistant that manages the current location re-REGISTERs the user with the SIP proxy server, so that any incoming call for that user, the site assistant will first be notified by the proxy server and then depending on the call importance (i.e. emergency calls), the site assistant can decide to forward the call to the corresponding use, forward the call to the user voice messaging system or simply reject the call.

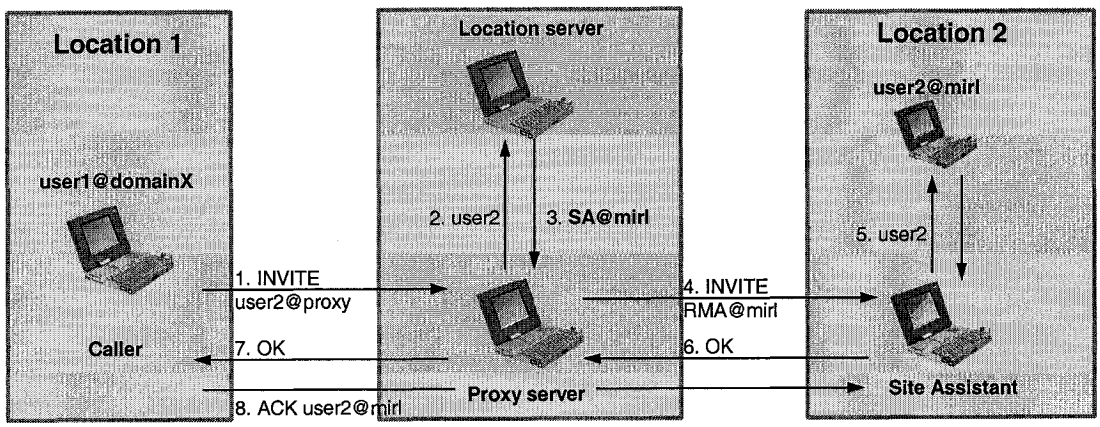

Figure 25. SIP RE-Registration

Figure 25 shows a scenario where user1 from location 1 calls user2 via the proxy server. The location server informs the proxy server that user2 could be reached using contact address SA@mirl which is actually the address of the site assistant at location 2 as the site assistant registered user2 on her/his behalf at the time arrived to location 2. All calls coming to location 2 are therefore intercepted by the local site assistant that then uses appropriate policies to make decision depending on each call category.

4.2.3 The Policy Service Agent (PSA)

As discussed in chapter 3, the Policy Service Agent (PSA) is an entity that monitors policies that are related to one application, where the Policy Management Agent (PMA) is an application-independent. Using the PMA the policy administrator needs only to specify high-level policy rules to achieve application policies. These policies are captured by the PSA and then deployed on individual agents where high-level policy rules are translated into agent specific policies to be enforced in application agents.

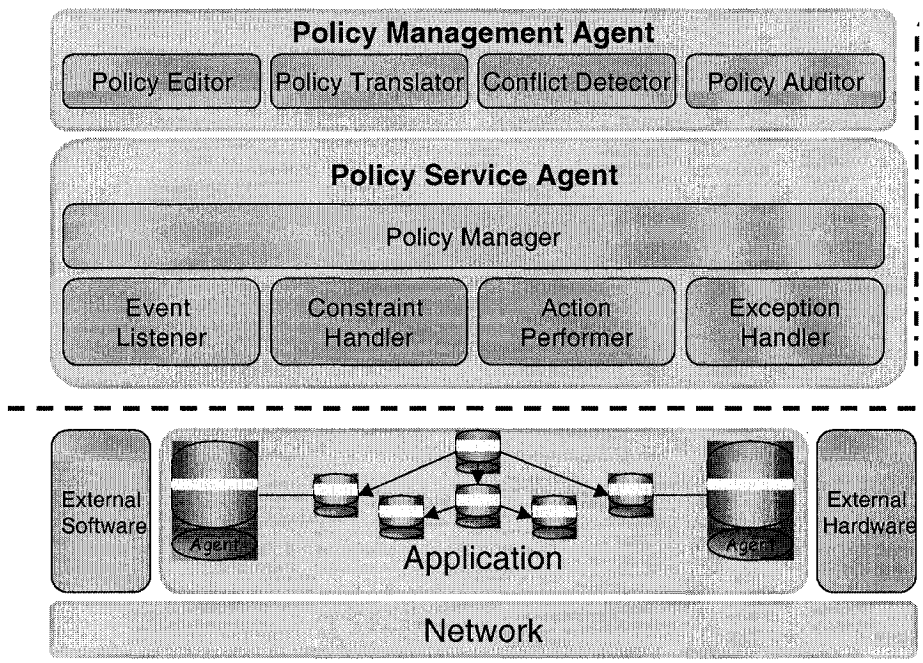

Figure 26. Policy Management and Service Agents

Figure 26 shows the internal architectures of the PMA and the PSA. The Policy Editor component provides the administrator with a view of the architecture of the application using an editing tool as shown in figure 27. The architecture is retrieved from the Application Profile along with the roles of every agent in the application. Once an agent is selected, a set of policies associated with the agent appears in the Policy Browser. Then the administrator may select a policy rule to view, update or delete the rule.

The Policy Conflict Detector deals with static conflict detection among policies to guarantee that the policies stored in the PIB are free of conflict. Policy Auditor allows the administrator to add audit options to the policies so that information about enforcement by the PSA could be retrieved when needed. The policy auditing is of particular interest in applications like Network Management where audit is a key feature. Hence for policies for which the audit is enabled (Yes in the Policy Audit field) information about which policy has been triggered, under which conditions, at what time, and at which node is then stored.

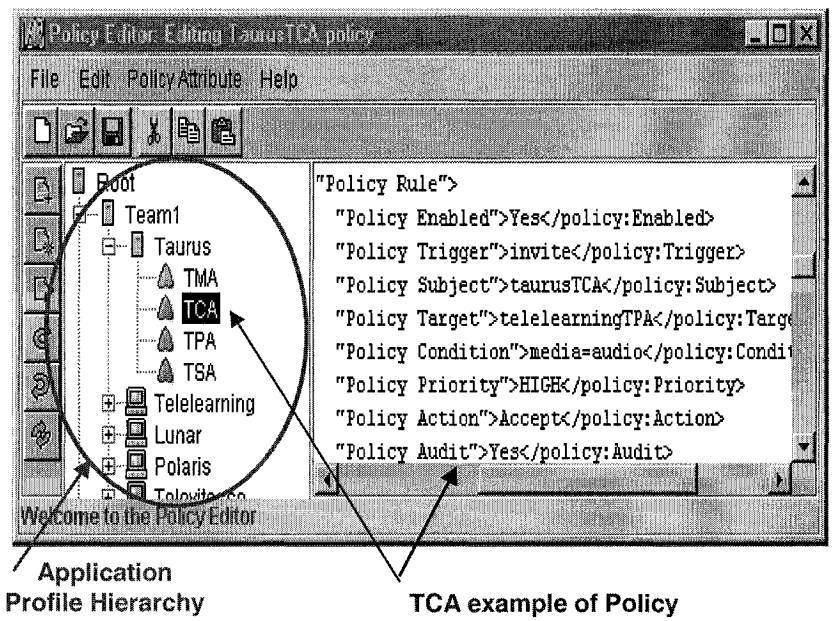

Figure 27. A sample of editing application policies

The Policy Translator is in charge of translating high-level policies to low-level policies to be used by the PSA.

The PSA is composed of the Policy Manager that maintains the translated policies and coordinates their execution by monitoring the other components of the PSA. The Event Listener listens for events relevant to a policy. As soon as an event is intercepted the information is transferred to the policy manager to check the policies and then the Constraint Manager determines the policy or a list of policies for which the conditions are satisfied. Among these policies, the policy manager determines the actions to be invoked by the Action Performer. The Exception Handler looks over the

outcome of the actions and reacts to possible exceptions.

4.2.4 The Policy Generator

As mentioned earlier, the policy generator has the role of making the framework and therefore the implemented applications context-aware. It produces specific policies depending on the analysis of contextual information at the site a mobile user is visiting. The produced policies have the same format as policies edited by the policy management agent, but the format of the input (i.e. contextual information) may differ from one site to another depending on the entity responsible of managing the context at each site. These entities are not part of this thesis. It is assumed that there is an entity that collects and interprets the context at different sites and these entities are capable of communicating with mobile applications via the policy generator.

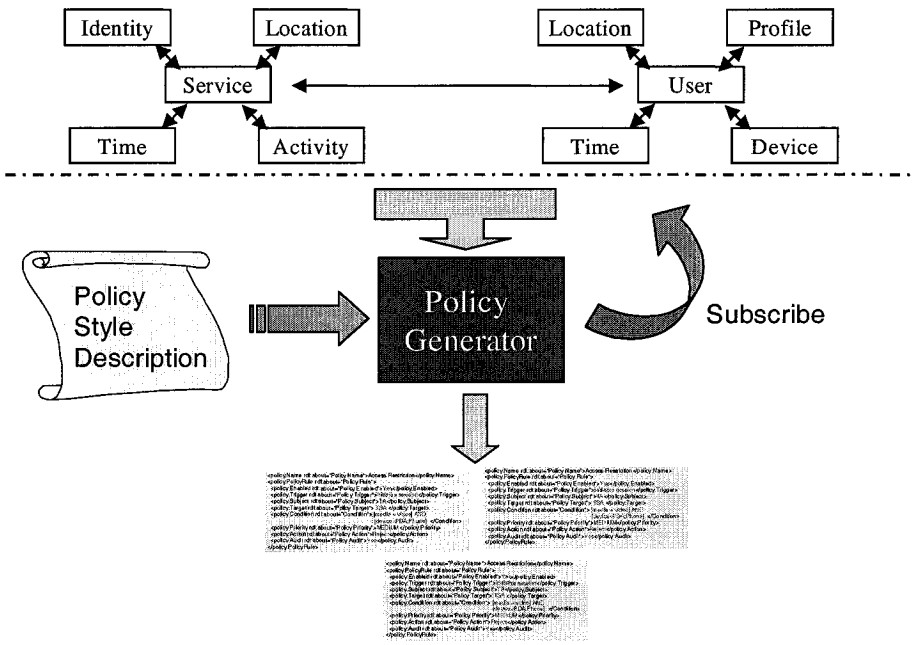

Figure 28. the Policy Generator

As shown in figure 28, the work of the policy generator is mainly divided into two parts: (i) subscribing to the context aware entity that is in charge of acquiring and modeling the context at the visited location, (ii) creating one or more specific policies after being notified by the context aware entity using the policy style description.

Table 5 shows an example of a context policy that is used in our ad hoc group meeting application prototype, which is described in chapter 5. The policy states that the agent PSA will notify a specific user (i.e. Eric) when the on going activity will be meetingA and the video source is VideoServer1, and then authorizing him to use the video streaming service. The user will be automatically authenticated, as he will be detected and recognized by the context management entity (i.e. CMA) while entering the MMARL lab. The policy is only valid to MMARL-Group members, and applies only during the time interval 12:00PM till 14:00 PM on Monday, May 24th.

Our approach in modeling and representing policies enriched our agent framework with the ability of common understanding of policies as well as the ability to process, derive and trigger new policies. For instance, using our policy generator, this policy can be applied to location B502, which is not stated in the policy. This new generated policy is also valid because B502, as a location, is found to be equivalent to MMARL location. Policies that apply to MMARL are perfectly legal to B502 and are automatically assigned to this location with no human intervention.

Table 5. An example of context policy generation

Policy	Antecedent 1	Antecedent 2	Antecedent 3
<Obligation rdf:ID= "VideoStreaming"> <appliesWhen> <ConTime:FormalType rdf:ID="ClockMeeting"> <ConTime:hasTimevalue> 12:00PM,14:00PM </ConTime:hasTimevalue> </ConTime:FormalType> </appliesWhen> <appliesWhen> <ConTime:StandardTime rdf:ID="CalanderMeeting"> <ConTime:hasTimevalue> Monday,May 24, 2004 </ConTime:hasTimevalue> </ConTime:StandardTime> </appliesWhen> <appliesTo> <Confoaf:Group rdf:ID= "MMARL-Group"> <Confoaf:name> MondayGroupMeeting </Confoaf:name> </Confoaf:Group> </appliesTo> <enforcementType> <PolicyEnforcement rdf:ID="Positive"/> </enforcementType> <appliesWhere rdf:resource="&Loc;MMARL"/> </Obligation>	<ConRule:Antcedent rdf:ID="polAntc1"> <ConRule:EntityA> <Voc:Variable rdf:ID="VideoSource"/> </ConRule:EntityA> <ConRule:entityB> <ConDev:Server rdf:ID="VideoServer1"/> </ConRule:entityB> <ConRule:relation> <Voc:Property rdf:ID="is"/> </ConRule:relation> </ConRule:Antcedent>	<ConRule:Antcedent rdf:ID="polant2"> <ConRule:EntityA> <Voc:Variable rdf:ID="DestinationUser"/> </ConRule:EntityA> <ConRule:relation rdf:resource="#is"/> <ConRule:entityB> <Confoaf:Person rdf:ID="Eric "/> </ConRule:entityB> </ConRule:Antcedent>	<ConRule:Antcedent rdf:ID="polant3"> <ConRule:EntityA> <Voc:Variable rdf:ID="Activity"/> </ConRule:EntityA> <ConRule:relation rdf:resource="#is"/> <ConRule:entityB rdf:resource="&Time; meetingA"/> </ConRule:Antcedent>
	Consequent 1	Consequent 2	
	<ConRule:Consequent rdf:ID="polcons1"> <ConRule:EntityA> <Voc:Variable rdf:ID= "ProvideService"/> </ConRule:EntityA> <ConRule:relation rdf:resource="#is"/> <ConRule:entityB> <ConProf:ServiceCategory rdf:ID="VideoPrinter "/> </ConRule:entityB> </ConRule:Consequent>	<ConRule:Consequent rdf:ID="polcon2"> <ConRule:EntityA> <Voc:Variable rdf:ID= "AuthenticateVia"/> </ConRule:EntityA> <ConRule:relation rdf:resource="#is"/> <ConRule:entityB> <Confoaf:SoftwareAgent rdf:ID="CMA-MMARL"/> </ConRule:entityB> </ConRule:Consequent>	

4.2.5 The Site Assistant

The Site Assistant is responsible of managing intra-site and inter-site communications using ACL (figure 29). It assists the visiting user at the visited location by performing the following tasks:

- Determine a temporary user profile at the visited site: the site assistant establishes a link with the site assistant at the home location and monitors a negotiation process to achieve an agreement for the user profile that includes the role of the user and authorized services.
- Provide access to the authorized services that could be either local services, home services or provided by an ISP.
- Act as an event server that notifies appropriate agents depending on the situation of use. For instance, when a new user enters a conferencing room, the site assistant notifies the SIP wrapper of her presence so that she will be invited to join the main conference.

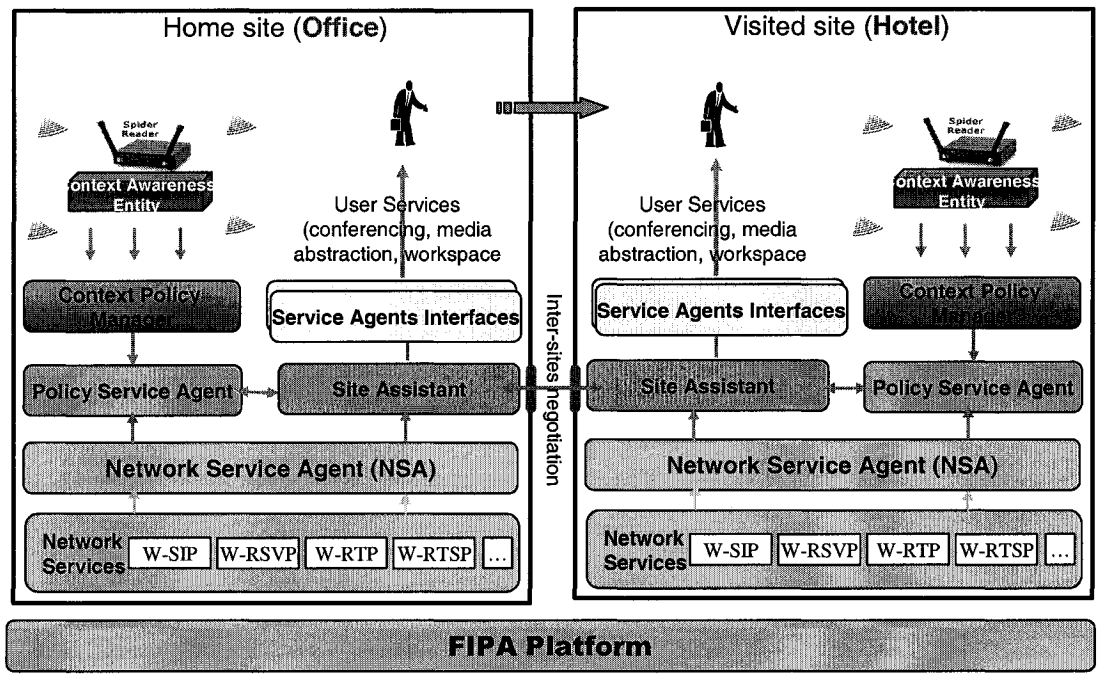

Figure 29. Inter-sites communication via the Site Assistant

4.2.6 Service Agents

Services to which mobile users could access while at a visited site could be either provided by agents (agent-based services) or by other entities (non-agent systems). Service agents establish relationship between agents and other software systems. They define generic interfaces that allow agents composing an application to communicate and interact with non-agent software systems such as databases, speech synthesis programs and media abstraction systems. Each service agent encapsulates the complexity of the underlying service to the agent-based application by translating the commands contained in ACL messages into operations on the underlying software system. A service agent allows an agent-based application to:

- Request a dynamic connection to an external software system,
- Invoke operations on the software system,
- To be informed of the results of operations,
- To query the properties of the software system,
- Set the parameters of a software system,
- Subscribe to events of the software system,
- Manage the state of the service,
- Terminate the service.

Figure 30, as described in FIPA specifications [99], shows a generic service agent that wraps a legacy system. A service agent is usually composed of an internal part which provides a dedicated mapping to the corresponding legacy system using specific mechanisms such TCP/IP protocol, IIOP or RMI, and an interface to the agent technology to allow agents' interaction using ACL messages.

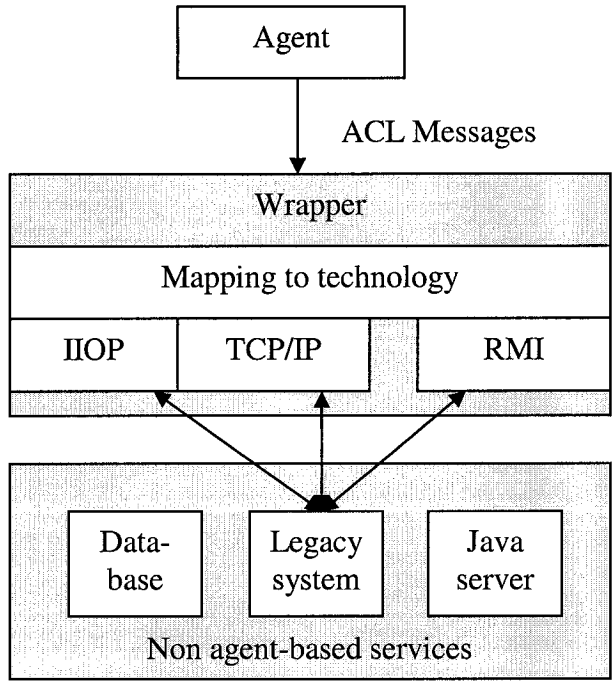

Figure 30 Generic Service Agent Model Wrapping a Legacy System

Service agents are also referred to as wrapper agents or service gateways. There is no significant difference between service agents, wrapper agents as they wrap non-agent systems so that they interact with agents' community, and service gateways as they encapsulate the complexity of dealing with underlying service.

Multiple service agents have been implemented within this thesis as it is shown in the next chapter.

4.3. Agent Mobility

As mentioned in chapter 2, section 2.3, FIPAOS which is the platform we chose for supporting the framework agents, does not provide agent mobility. Agent mobility has been introduced in FIPA specifications, namely the "Agent Management Support for Mobility" [99], but not implemented and integrated to FIPAOS –open source- platform.

FIPA specification recognizes multiple ways of agent mobility including code and state mobility, agent migration and cloning, but defines two types of mobility

protocols: Simple Mobility Protocol and Full mobility Protocol. Each can apply in different situations of use.

Figure 31 shows the full mobility protocol where agent A takes the control of the move and not delegating it to the platform. The move is considered complete after the agent's code and identity have been successfully transferred, while in the simple mobility, the move is delegated to the two involved platforms (source and destination platforms).

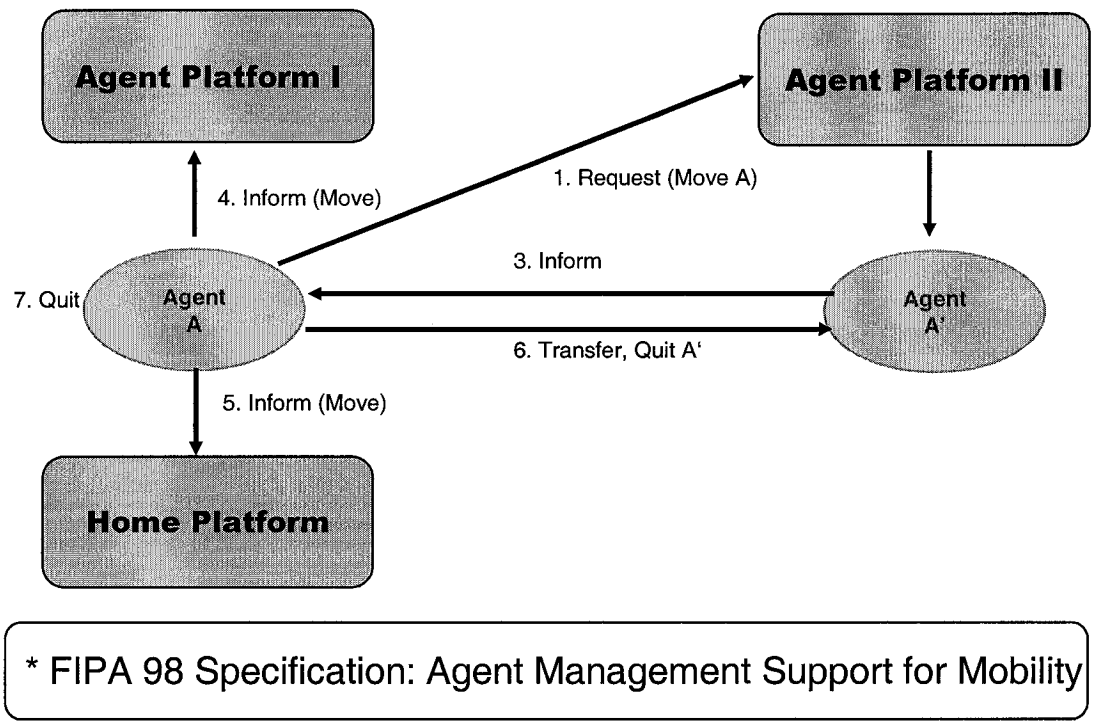

Figure 31 Full Mobility Protocol

There are four FIPA actions that are related to agent mobility: *move* and *transfer* that could be requested by a mobile agent to transfer its code and its identity respectively, and *execute* and *terminate* issued by the platform (i.e. agent management system) to execute an agent on the platform and to terminate its execution respectively.

To add agent mobility features to the framework, we introduced the Mobility Management Agent (MMA) which is responsible of agent mobility at each site. When an agent decides to migrate to another platform, it sends a move request to the MMA at that platform, and then the MMA ensures its migration according to the full

mobility protocol described above. Figure 32 shows the sequence diagram for moving an agent A from the platform P1 to P2 and then back to the platform P1 using the MMA:

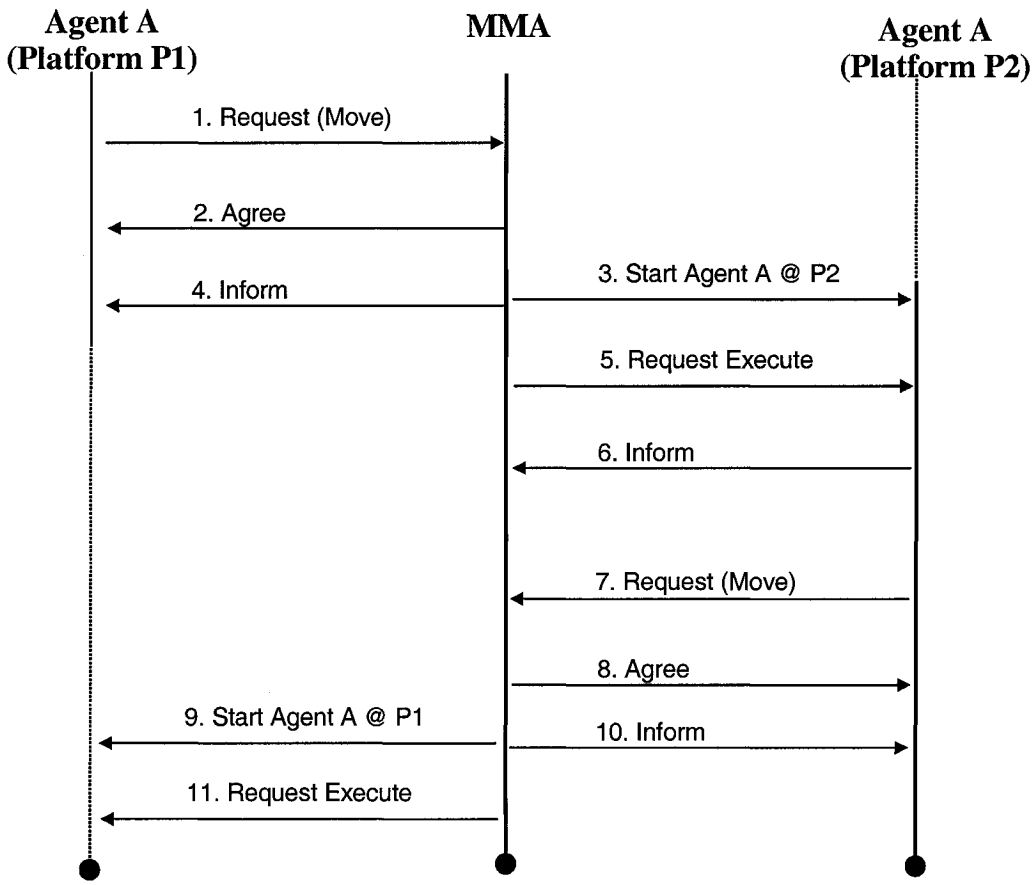

Figure 32. Agent Mobility and MMA

When an agent applies to move to a remote platform, a mobile agent description must be supplied with different attributes that characterizes the agent. An example of a mobile agent description in RDF format is given as follows:

```
<agent:name rdf:about="agent name">SideBarAgent</agent:name>
<agent:description rdf:about="description">
 <agent:location rdf:about="agent address">http://taurus/sba</agent:location>
 <agent:destination rdf:about="agent destination">laptop_ip</agent:destination>
 <agent:platform rdf:about="agent platform"> fp_ltop </agent:platform>
 <agent:language rdf:about="agent language">Java</agent:language>
 <agent:format rdf:about="agent format">bytecode</agent:format>
 <agent:code rdf:about="agent code">room.Sidebar</agent:code>
 <agent:protocol rdf:about="agent protocol">full-mobility</agent:protocol>
</agent:description>
```

The agent mobility features could be implemented as part of the agent management system (AMS) inside the agent platform, but for agent platforms interoperability and to allow the ability to use other agent platforms such as JADE, we preferred to keep the MMA as a part of the framework like all agents composing the framework that have been described in the above sections.

4.4. Applications Prototypes and Scenarios

Different scenarios have been developed to demonstrate the use of the framework architecture with various applications that make use of mobility and context-awareness. In the next chapter we describe three implemented scenarios using the framework:

- a scenario from the Multimedia Service Provisioning for Mobile Users application,
- a scenario from the Management of Virtual Teams application,
- and a scenario from ad-hoc group communication system.

CHAPTER 5

IMPLEMENTATION AND APPLICATIONS PROTOTYPES

This chapter describes the implementation of the main components of the framework and its use in supporting mobility and context-awareness in mobile environments by implementing three different applications prototypes to demonstrate various features of the framework.

Figure 33 depicts the main phases in the implementation process and prototyping. The first phase focused on the policy modeling and the agent model that integrates its own policies. The policy-based agent negotiation process has then been designed, followed by the framework agents described in the previous chapter. Various revisions have been applied to the framework as the development of the multimedia service provisioning prototype has been started.

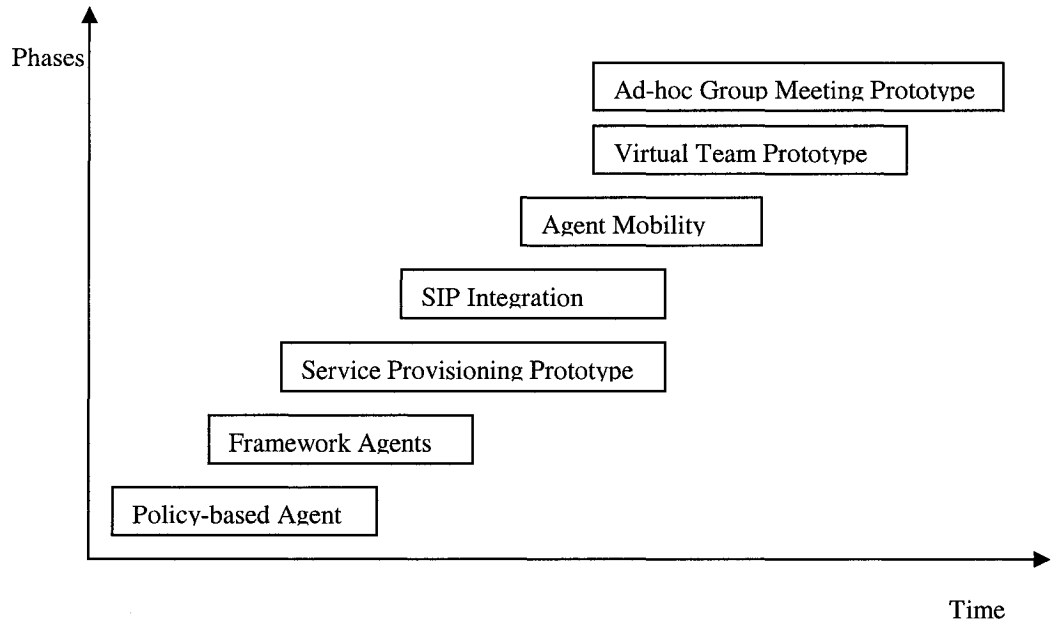

Figure 33 Main phases in the implementation process and applications prototypes

The framework agents have been implemented using Java, and supported by FIPA-OS agent platform. To make use of agent mobility, we added this feature to the agent platform by developing our own agent mobility module according to FIPA specifications as it is not provided by FIPA-OS open source. Two more applications prototypes have been implemented to demonstrate the applicability and the sweetness of the framework to support multiple applications in mobile environments.

5.1 Policy-based Agent Implementation

The proposed policy-based agent model has been implemented in two steps. The first step concerned the representation of policies. We used the resource description framework to fulfill this task and the second step was the integration of RDF policy file with its corresponding agent.

Figure 34 shows the UML representation of our agent model as described in chapter 3. A developed agent inherits basic properties from *FIPAOSAgent* class that provides methods for registration, messages sending, setting up tasks, getting platform profile and agent shutdown. It also contains a number of Task implementations that give the

functionality of the agent. A TaskManager coordinates tasks inside each agent by handling a set of events. The agent behavior is controlled by a set of policies that are monitored by AgentPolicy class which invokes *getPolicyFile()* method to load agent's policies and translates them into a decision tree for policy evaluation and enforcement. The policy file has the same name as the corresponding agent with the extension policy.

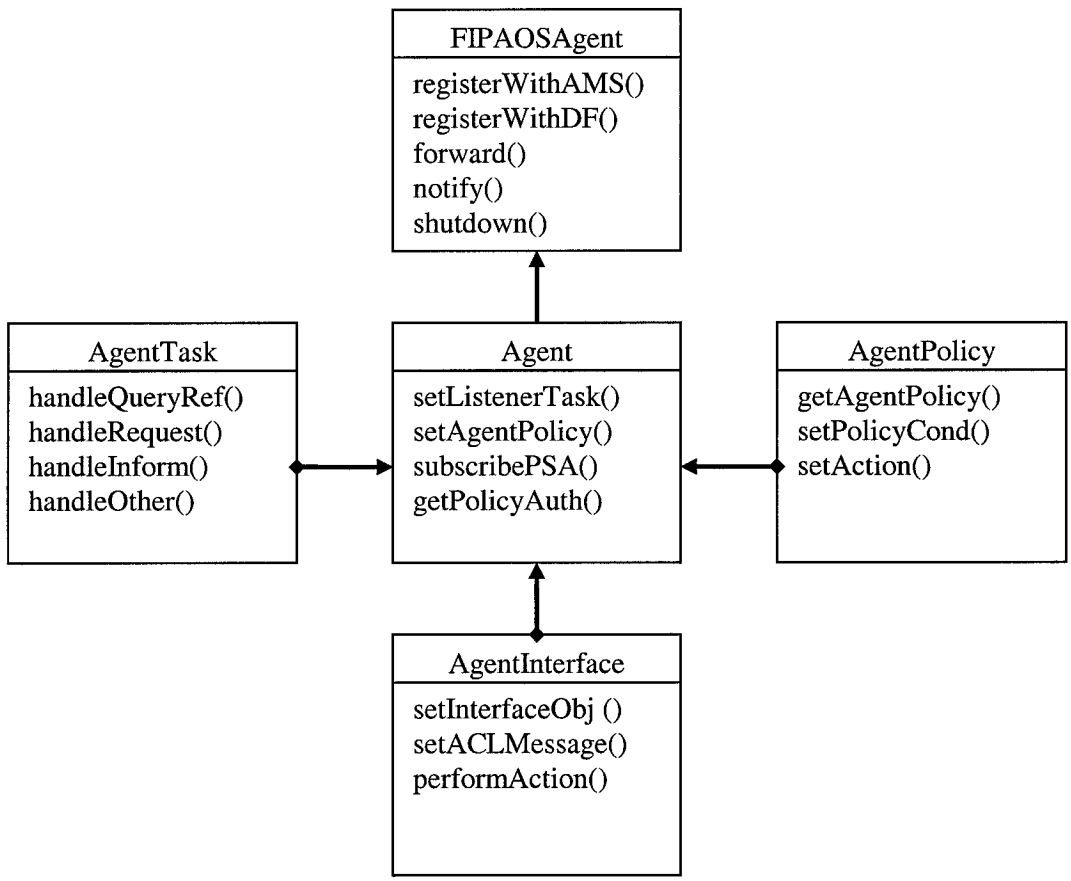

Figure 34 Class Diagram representing the framework agents

Figure 35 shows an example of an agent template that has been used in the implementation of the framework agents.

```
/* The Template Agent */
public class AgentTemp extends FIPAOSAgent
{
 public AgentTemp( String platform_profile, String name, String owner )
 {
     super( platform_profile, name, owner, true );
    // Create the agent Listener
    setListenerTask( new AgentTask() );
    // Create the agent policies
    setAgentPolicy( new AgentPolicy(agentName+".policy") );
    // Attempt to register with AMS
    try
      {    registerWithAMS();
       } catch (AMSRegistrationException amsre) { }
    // Agent Interface
    _aTemp = new AgentFrame(this, _srID, "agent template", "Agent Template Interface");
    _aTemp.setVisible( true );
 }
//---------- PUBLIC METHODS ----------
 public class AgentTask extends Task
  {
    public void handleRequest( Conversation conv )
     {
           ACL acl = (ACL) conv.getACL( 0 ).clone();
           System.out.println( "Recieved acl..."+acl );
           _aTemp.setACLMessage( acl );
    }
   public void handleInform( Conversation conv ){
        ACL msg = conv.getACL(conv.getLatestMessageIndex());
        System.out.println( "Recieved ACL:: "+msg );
       _aTemp.setACLMessage( msg );
 }
}
/---------- PRIVATE METHODS ----------
 private void psaPolicyAuth(String subject, String role, String action, String condition)
  {
    try{
       ACL acl = new ACL(FIPACONSTANTS.FIPA_REQUEST);
       acl.setPerformative(FIPACONSTANTS.REQUEST);
       acl.setSenderAID( new AgentID(rmaHost) );
       acl.setReceiverAID(new AgentID(psaHost));
       acl.setOntology("fipaos-request");
       acl.setContentObject(tuple);
```

Figure 35 A Sample of the framework agents' template implementation

5.2 Multimedia Service Provisioning Prototype

5.2.1 Introduction

The multimedia service provisioning prototype application describes a scenario where nomadic users moving from one site to another set are provided by a set of authorized services which reflect their pre-determined preferences available at their home sites. Based on agents' negotiation, users may have access to local services that are equivalent to those they have at home sites, or have access to their home services or ISP's. At a visited site, a user profile is therefore determined taking into consideration the privacy of the visited and the home sites information.

In developing the application prototype, we considered specific requirements necessary for an efficient characterization of mobile users and multimedia services that they may wish to use. We also considered the capabilities of the network and the devices, and the policies binding all these components together so that they consistently satisfy user needs. Such requirements include:

- Ability to create and manage a mobile user's profile that enables multimedia services to be fully personalized.
- Ability to access local services at visited sites using the personalized preferences available at the home site.
- Authentication of users and automatic configuration of authorized services to be performed at visited sites.
- Obtaining a seamless resource allocation when requesting remote access to a personalized multimedia service.
- Service presentation adapted to the device capabilities and the authorized quality of service.
- Control and management of dynamic changes in preferences, locations and activities that may occur during mobile users' travels via policies.

5.2.2 Participating Sites Scenario Description

This subsection describes how a temporary profile is determined for a visiting user, and how the user can use his/her personalized services. Our example is a user (Tom) from Mitel Corporation visiting the University of Ottawa site.

As shown in figure 36, the scenario involves three sites including the University of Ottawa, the National Research Council of Canada and Mitel Corporation. Each site maintains its private and specific policies and a list of local services that may be provided to mobile users.

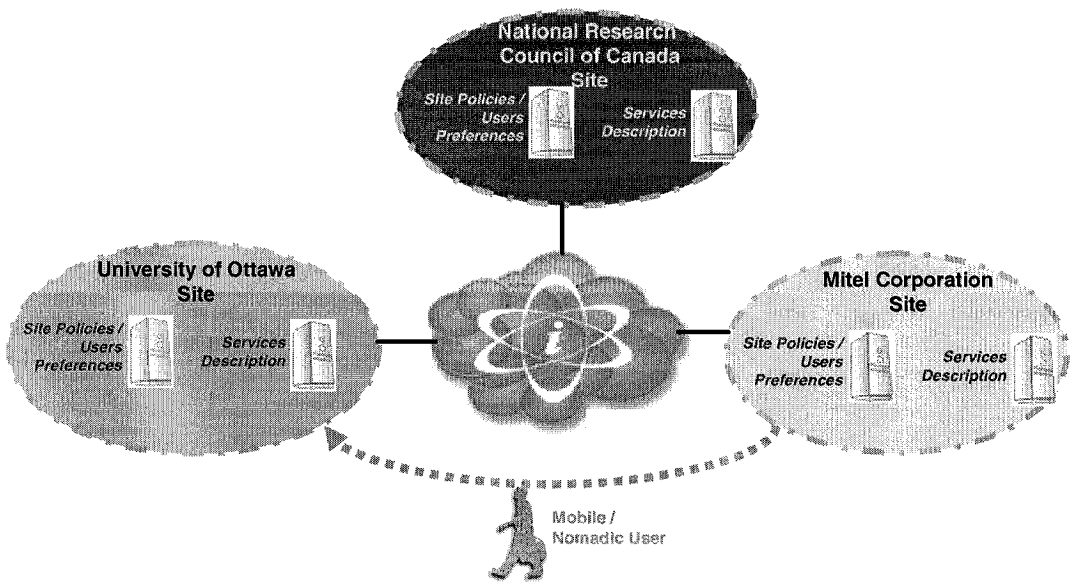

Figure 36. Three Sites Service Provisioning Scenario

The home profile of Tom includes his personal information (name, address, phone, url, ...etc), his role as a manager, and services to which he prefers to access to while moving within entities that are composed in a private network.

The preferences of Tom include currently two messaging services namely:

- Media Abstraction service (MediABS) which provides a video mail abstraction service by extracting consecutive frames required to detect the camera cut events;
- Call Forwarding Service (CFS) which selects the incoming calls at the home site and forwards the preferred calls to the user at the visited site.

5.2.3 *The Framework and the Prototype Implementation*

Figure 37 provides the sequence diagram of the negotiation protocol between the visited and the home site assistants looking for an agreement about the temporary profile.

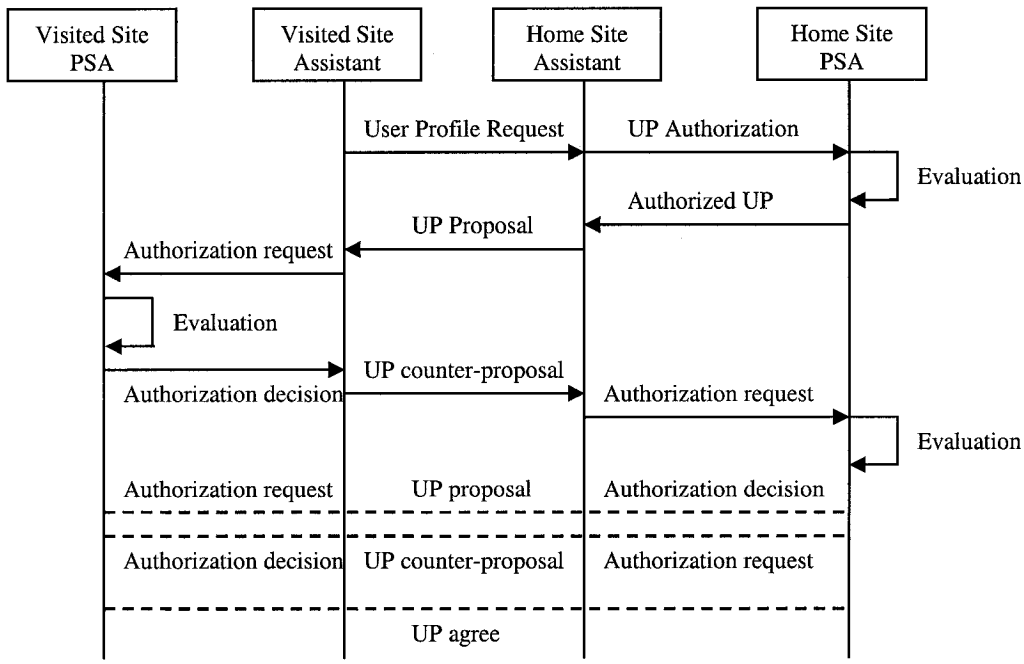

Figure 37. The Sequence Diagram for the Negotiation Protocol

Once a visiting user is authenticated at the visited site, the Home-SA gets the user home profile from the home repository and requests an authorization from the local policy service agent. The home-SA replies to the visited-SA with a proposal message. Upon receiving a proposal, the visited-SA evaluates the profile parameters and may request to update some of them according to its local policies. For example, the home-SA may propose that the visiting user would use two services: the *MediABS video* service with *cost = 10* units and the *category = video streaming* and the phone *Call Forwarding Service (CFS)* with *urgency = all*. According to its local policies, the visited-SA may generate a counter-proposal where access to CFS service is denied and the cost of MediABS service needs to be increased or the category decreased: [*MediABS (cost HIGH or category LOW), CFS (DENIED)*]. The home-SA needs

either to increase the cost or to lower the category. According to its local and private decision making, a new proposal will be submitted, until an agreement is achieved and the visiting user can now access the MediABS video service with parameters "*cost=15*" and "category=*video streaming*".

Because of the resource reservation process, the negotiated parameters may not be guaranteed at the time the user requests the service at the visited site. Each authorized service agent would then have to negotiate local resources such as time, space and bandwidth with the local Resource Agent (RA). In the case of MediABS service, the allowed space will determine the value of the *compression ratio* (as the percentage of the allowed space over the average size of the video in the negotiated category). The quality of service may even be reduced to a lower level (from video steaming to color key frames) to match the available resources and the capabilities of the device from which the visiting user logged on. For instance, if the total size of the video file is about 3.5Mb, but the allowed space is limited to 320Kb, the quality of service will be reduced to color key frames with a total size of 240Kb. The compression ratio in this case is about 30%.

If the user disconnects before the service agent sends back the results, the service agent stores the retrieved results in the XML format. The results then become available for presentation the next time the user logs on (figure 38). As depicted, the XML result file contains four main parts namely: user, service, request and result as we mentioned.

Lines 5-8 present the elements used within the user part. They include the identity that uniquely describes the user who submitted the request, title or affiliation of the user, the visiting site of the user from where he submitted the request and his/her home site. Lines 11-24 show the elements used within the service part that are the name of the service, its location, the cost attribute and the accepted level of the quality of service and the compression ratio.

The request part is illustrated in lines 27-33. These parameters include the request unique reference number, the name of the media file that has been processed, the start and end points of the media segment within the video, the name of the operation used to process this request, the date and time of the request.

Lines 36-54 of the output report depict the result part of the request. The main result parameters are the CPU processing time for performing the request, the total number of video segment frames, the number of key frames, the frames references within the media file, the total size of key frames, the format of the frames, the protocol that could be used to retrieve and browse this report and the server address of the frames files.

```
(1)   <?xml version="1.0" ?>
(2)   <!DOCTYPE report SYSTEM "KF_report.dtd">
(3)   <report>
(4)         <user>
(5)           <id>xyz@sol.genie.uottawa.ca</id>
(6)           <title>PhD student</title>
(7)           <vlocation>Mitel</vlocation>
(8)           <hlocation>UoOttawa</hlocation>
(9)         </user>
(10)        <service>
(11)          <name>MediABS</name>
(12)          <location>UoOttawa</location>
(13)          <parameter>
(14)                <key>cost</key>
(15)                <value>15</value>
(16)          </parameter>
(17)          <parameter>
(18)                <key>quality-of-service</key>
(19)                <value>colorKeyFrames</value>
(20)          </parameter>
(21)          <parameter>
(22)                <key>Compression-Ratio</key>
(23)                <value>30</value>
(24)          </parameter>
(25)        </service>
(26)  <request>
(27)          <reference>624</reference>
(28)          <input>video122.avi</input>
(29)          <startframe>1/4</startframe>
(30)          <endframe>3/4</endframe>
(31)          <operation>key framing</operation>
(32)          <submitdate>17_Nov_2004</submitdate>
(33)          <submittime>14:26:47 </submittime>
(34)  </request>
(35)  <result>
(36)          <processing>ok</processing>
(37)          <time>8.19 s</time>
(38)          <initial_size>2.310 Mb</initial_size>
(39)          <totalframes>295</totalframes>
(40)          <processedframes>28</processedframes>
(41)      <frames>
(42)          <number>4</number>
(43)          <fvalue>73</fvalue>
(44)          <fvalue>120</fvalue>
(45)          <fvalue>185</fvalue>
(46)          <fvalue>221</fvalue>
(47)          <total_size>18.428 kb</total_size>
(48)          <format>JPG</format>
(49)          <dimension>160x120</dimension>
(50)      </frames>
(51)      <dlocation>
(52)          <protocol>http</protocol>
(53)          <host>altair.genie.uottawa.ca</host>
(54)          <path>\mediabs\outputs\624_result\</path>
(55)      </dlocation>
(56)    </result>
(57)  </report>
```

Figure 38. An example of a result report for the video service in XML format.

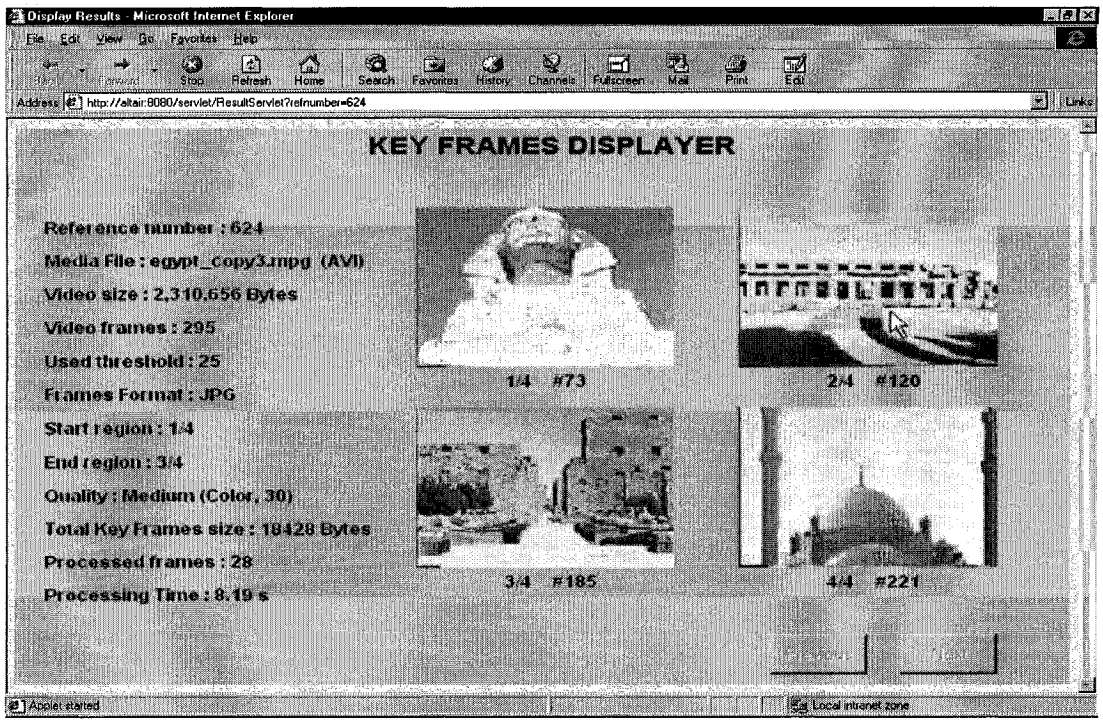

Figure 39. A Sample of the user interface browsing tourist destinations in Egypt.

The output result quality lies in three categories: the first category is to use the video streaming with less content than the complete stream. The second category, as shown in figure 39, is to use color and high quality JPG image files (i.e. with low compression ratio parameter) The third category is to provide only a gray scale version of these key frames with low JPG quality (i.e. with high compression ratio parameter). A ratio parameter CompressionRatio is determined to guide the selection of the output result within each category as shown in figure 40. If the results are in the form of key frames, the service will use this ratio to generate lossy JPEG [104] key frames that has encoding compression rate relative to this parameter. Otherwise, in the case of video streaming result, the service will use this parameter to generate, using a video editing utility such as Video Edit Magic [105], a new trimmed video version of the original video but with dropping frames (i.e. proportional lower frame rate). The new video frame rate will be proportional to that compression ratio parameter for the same video segment so that the audio stream will have the original sound quality.

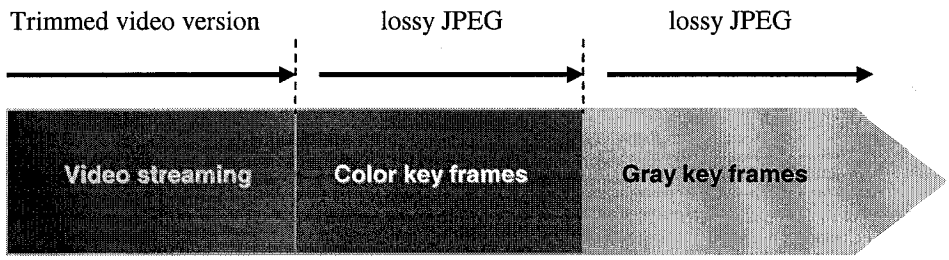

Figure 40. Output results categories and Compression ratio parameter.

In the next subsection we present an overview of the media abstraction service and its integration via its corresponding service agent into the prototype.

5.2.4 Media Abstraction Service

The number and variety of the end devices that can be used by mobile users made it necessary to adapt the presentation of multimedia services. Research continues into the standardization of location-based mobile access to multimedia content. One example is the MPEG-7 working document on mobile requirements and applications [68] from the Moving Picture Experts Group (MPEG). They are currently working on enabling mobile access to MPEG-7 information using GPS, as well as context-aware and location-dependent technologies. A user would be able to watch the trailers of current movies from the nearest cinema by wireless connection while he is walking or driving his car.

A stand-alone media abstraction service to perform media indexing, summarization and conversion has been designed and developed in our laboratory as part of a separate work [68]. The service includes image browsing, video streaming, content-based analysis, and text to speech converters. Specifically, a video segmentation algorithm has been developed based on color histograms and using an opportunistic binary penetration technique [68].

The service addresses the problem of analyzing multiple video formats. It aims to provide video summaries by accurately identifying the different cut changes within a video segment. The algorithm defines various configurations for the extracted frames, which could be in color or gray, with different image qualities and formats to match

the capabilities of the users' devices.

This service exploits only the two most significant RGB bits of each color. To do this, it uses a masking operation, which evaluates every two-frame histogram difference. If the average difference exceeds a defined threshold, the two frames are said to represent an abrupt camera cut. However, in some cases, the color distribution information may not detect the cut. This problem can be solved because each frame is partitioned into a number of disjoint blocks, and makes use of the information available in the histogram distribution.

Figure 41 shows the subsystem of media service as a black box and indicates its interface to the corresponding service wrapper agent. The process accepts an input request file from the service agent, supplying the parameters of the request. It generates a result report containing the detected key frames along with the confidence percentage of each key frame. Then the service agent reformats the report, using XML as mentioned, before passing it to the user.

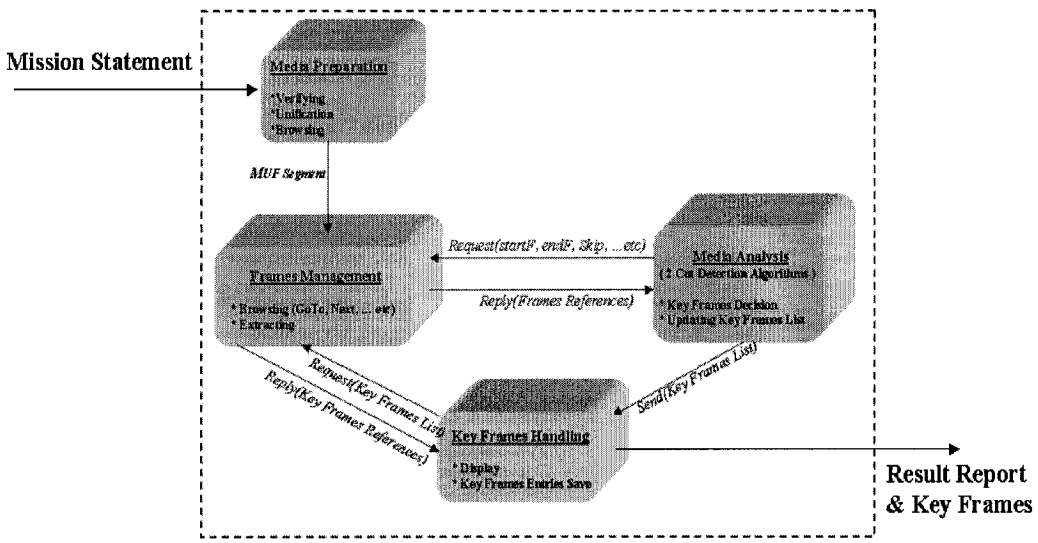

Figure 41. The use of the media abstraction process as a service for mobile users.

Figure 42 shows an example of the mission statement input in XML format. The service agent generates this file according to the negotiation process. The agent provides the request to the service process itself to execute and complete the mission.

```
<?xml version="1.0" ?>
<mission>
    <ReferenceNumber>624</ReferenceNumber>
    <InputMediaFile>video122.avi</InputMediaFile>
    <UserID>xyz@sol.genie.uottawa.ca</UserID>
    <iDate>28/9/2001</iDate>
    <iTime>4:25:00PM</iTime>
    <StartF>1/4</StartF>
    <EndF>3/4</EndF>
    <TemporalSkip>5</TemporalSkip>
    <SpatialSkip>5</SpatialSkip>
    <Threshold>25</Threshold>
    <Operation>KeyFraming</Operation>
    <KeyframingMethod>6MSB_Blocks</KeyframingMethod>
    <PerformanceMethod>Binary_Penetration</PerformanceMethod>
    <NumberOfBlocks>9</NumberOfBlocks>
    <WorkingDirectory>vb5\for_integration\outputs</WorkingDirectory>
    <FramesFormat>JPG</FramesFormat>
    <ColorGrey>Color</ColorGrey>
    <CompressionRatio>30</CompressionRatio>
    <InitialFormSize>Normal</InitialFormSize>
</mission>
```

Figure 42. An example of video service mission request in XML format.

5.2.5 Discussion

We demonstrated the use of the framework by integrating the media abstraction service (MediABS), implemented in our laboratory as a stand-alone application. The prototype described above has been developed by implementing the service agent that represents MediABS service. The framework agents have been successfully reused to provide mobile users by temporary profiles and services access at visited locations. The context-awareness part was not used in this prototype according to the requirements of our partners for this prototype which is part of the Mobile Alliance Project involving Mitel Corporation, NRC and the University of Ottawa. Examples of policies that have been used within this prototype are given in tables 3 and 4 in section 3.2.1.

5.3 Virtual Team Application Prototype

5.3.1 Introduction

The globalization of markets and business processes have facilitated the evolution towards lightweight "virtual organizations" where a number of complementary organizations join together to tackle projects that they may not be able to perform individually. Lipnack & Stamps [59] define a virtual team as "a group of people who interact through interdependent tasks guided by common purpose" that "works across space, time, and organizational boundaries with links strengthened by webs of communication technologies". A virtual organization can be defined as a geographically distributed organization whose members are bound by a long-term common interest or goal, and who communicate and coordinate their work through information technology.

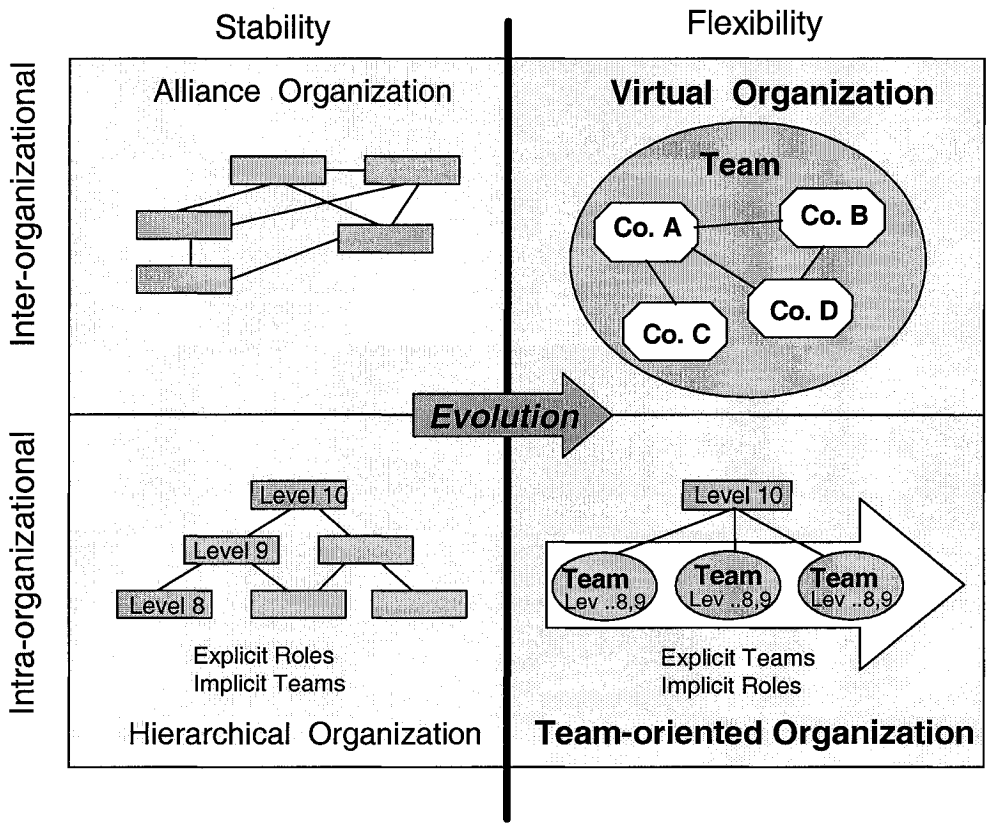

Figure 43. Hierarchical and team-oriented organizations

As shown in figure 43, emergent virtual organizations tend to have little regard for traditional hierarchical structures towards decentralized organizations that facilitate lateral communication through information technologies.

A virtual team consists of a dynamic group of individuals that do not necessary belong to the same organization. A virtual team also requires synchronous and asynchronous collaborative services such as conferencing, and requires coordinated network facilities that ensure that resources are shared flexibly and securely.

We have developed V-Team, a working prototype for the management of virtual teams. V-Team is designed and implemented to support multimedia and multiparty applications for geographically distributed teams across heterogeneous networks and devices. It is a prototype application that integrates the necessary team and network services as shown in figure 44.

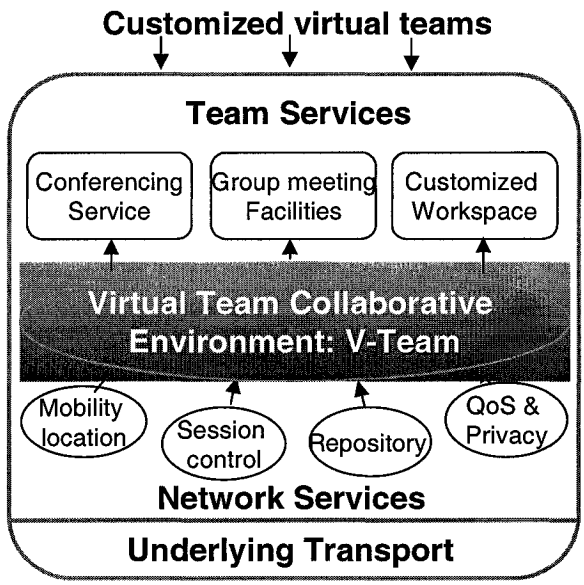

Figure 44. V-Team System Basic Model

Network services include the ability to track the locations of mobile team members, the quality of service, and the multimedia session control protocols for the management of collaborative sessions. V-Team removes the need for users or applications to deal with network services directly. It is also motivated by its ability to

support and manage the dynamic change that may occur in the structure and location of distributed teams, or their associated context across time and space.

Context-awareness allows V-Team to adapt team behavior to a wide range of uses. As a result, team services and applications have broader range, and great potential to be customized.

To facilitate the development of V-Team prototype, we made use of the framework software agents. To setup a virtual team session, we define a virtual meeting place to which is assigned a Team Assistant (TA) responsible for the place's control and monitoring. The TA corresponds to the Site Assistant that has been introduced in the framework. To locate and involve participants, the TA uses the Session Initiation Protocol (SIP) that has been already integrated to the framework. The TA customizes each team participant's environment by selecting information and services that correspond to the participant's privileges, the capabilities of the end-use device and existing policies at the location from which the participant logs on.

5.3.2 Scheduling and Setting up a Collaborative Session

In the following, we provide a scenario that describes the V-Team approach and, in particular, highlights features that have already been implemented as a prototype. Since the behavior of the system prototype is aimed at being completely monitored by policies via the Policy Service Agent (PSA), the administrator, using the PMA, defines policies that determine who can create virtual teams and logical resources that are available to team leaders.

The virtual team creation is customized according to the team leader privileges (e.g. resources assigned to the team, role, media and QoS assigned to each member). The team leader plays the role of the team controller, but can assign other team members as team controllers. The team controller has access to the TA agent, which provides collaborative session facilities.

Each team member is represented by a Personal Assistant, which has the ability to act on his behalf in conformance to a set of policies and may interact with him if no policy can be triggered.

At scheduling phase, the team leader may not be aware of participants' availability. PAs engage into a negotiation process to determine an appropriate schedule (*time, date, duration and frequency*) under the control of the Team Assistant playing the role of a mediator. Interactions among PAs are performed in exchanging information messages using ACL.

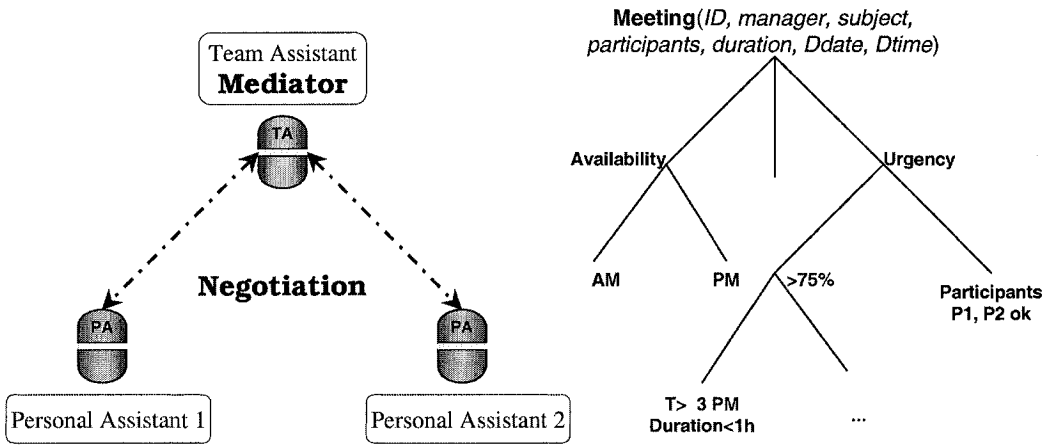

Figure 45. Team Assistant Scheduling Scenario

The negotiation starts by the team assistant producing a proposal according to the scheduling policies given by the team leader. The proposal is then sent to the PAs, which evaluate the proposal and possibly generate counter-proposals till an agreement or the deadline of negotiation, have been reached. Proposals and counter-proposals are based on time, date and duration parameters as shown in figure 45 scenario. At the time of setting up a collaborative session, team members are invited to join the collaborative session using SIP invitation (INVITE message). The W-SIP, which is part of the framework platform, sets up the SIP UA parameters automatically and waits for an INVITE response. When a team member is located by the SIP proxy, the corresponding SIP-UA sends a request to the PA that represents this member and waits for a response. Depending on the enabled PA policies, the response could be either ACCEPT to accept the call invitation, REJECT, BUSY or FORWARD as detailed in section 4.1.2. The response is then sent back to the W-SIP with the agreed SDP description. Context policies use these capabilities as conditions to trigger specific actions while in a particular context.

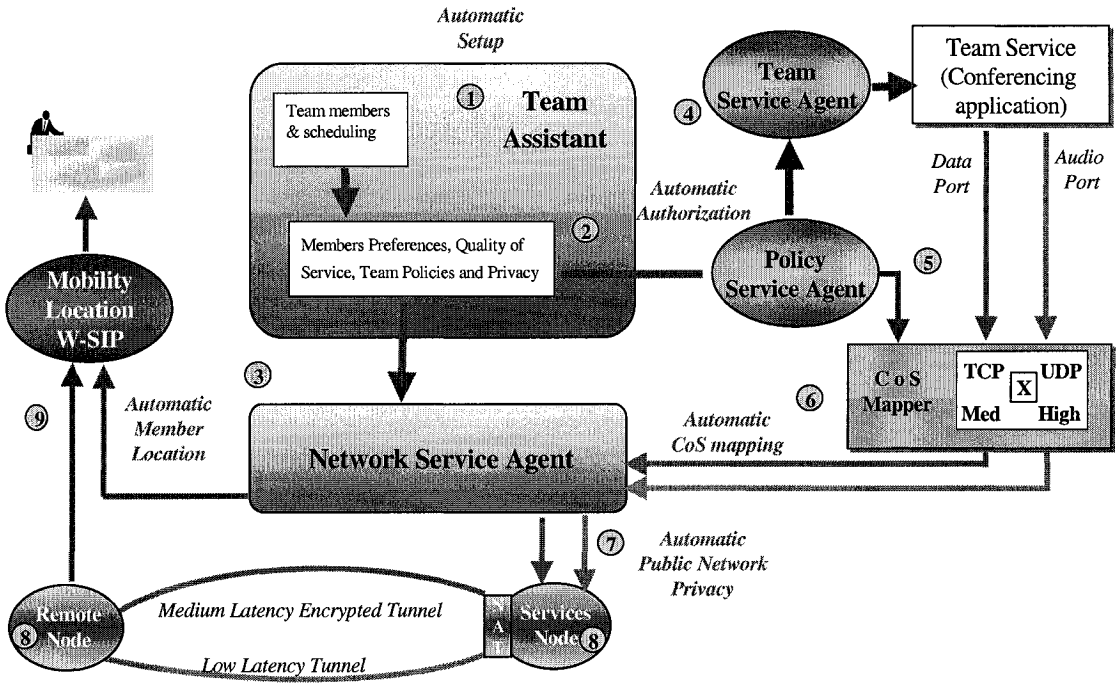

Figure 46. Example of V-Team behavior

Figure 46 shows the main steps that describe the process of setting up a pre-scheduled distance group meeting session among team members.

1. The Team Assistant automatically triggers a setup action by enforcing a specific context policy when the time-period occurs.
2. The Team Assistant gets the team members information maps their preferences with the authorized team services, according to their role.
3. The V-Team relies on SIP to locate and track members' mobility. The Team Assistant establishes a connection with the local W-SIP agent via the NSA, so that the authorized team members could be invited using SIP.
4. Members can now join the pre-scheduled conference, after the Team Service Agent (TSA) would have negotiated with the conference service to reserve the necessary resources. TSA are used in the same way as Service Agents discussed in the framework.
5. In this example, the audio component has been set up to receive high priority, low latency transport service while the data component receives medium latency.

6. The Class of Service Mapper links the appropriate CoS processing to the voice and data streams.
7. The NSA interacts with underlying transport mechanisms to ensure that the necessary resources for CoS and Privacy are secured and reserved. Context policies may request tunneling and/or encryption on Network Service Agent facilities.
8. The NSA then identifies the nodes that members are attached to and establishes the appropriate CoS and privacy links between them.
9. The TA launches the user agents' interfaces at the authorized participants' devices and displays interfaces with virtual places, members' status, team services and tools for leaving and/or joining a virtual place. A virtual place corresponds to the pair virtual team and session. It represents a place where virtual team members meet during one session.

A sample conferencing scenario is depicted in figure 47. In this example, participants collaborate using audio and video conferencing tools developed at the University College London (RAT and VIC respectively).

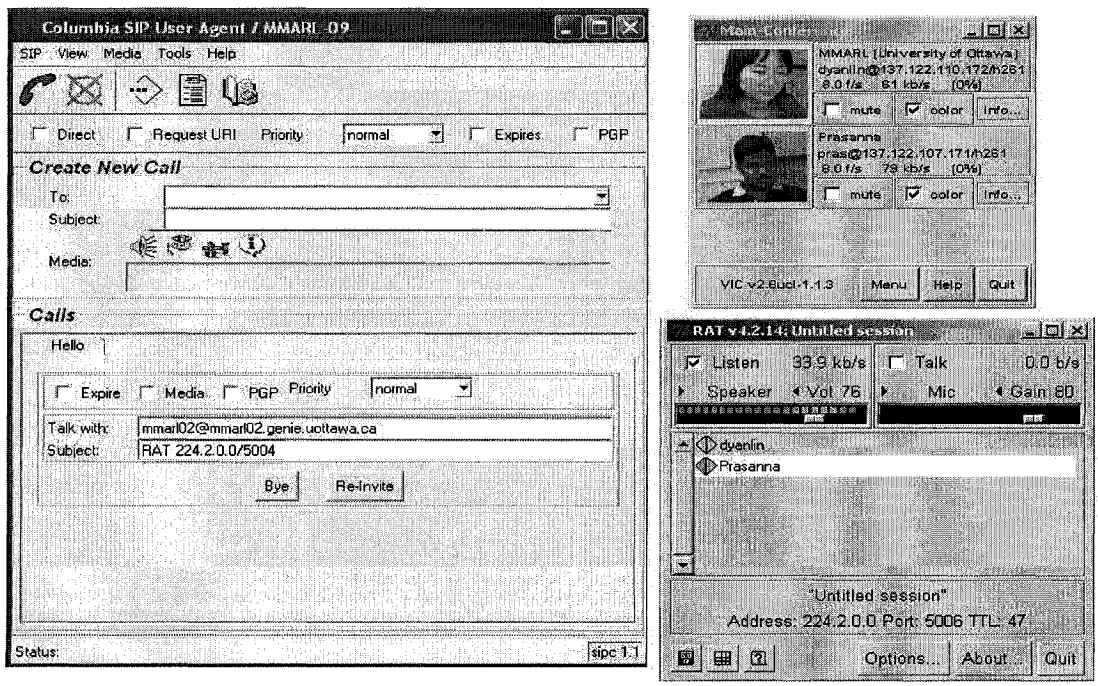

Figure 47. A sample of conferencing scenario with audio and video tools

5.3.3 Discussion

In the above described scenario, we focused mainly on the network service agent proposed in our framework. The integration of the session intitiation protocol has been successfully tested and effective audfio and video conferencing sessions have been demonstrated using SIP. More network services need to be integrated to the network service agent, so that it can efficently relieve developers from dealing with them directly.

The integration of RSVP protocol was also tested to allow the NSA reserve resources for the negotiated QoS and time intervals via RSVP. This integration has been done in a separate prototype that could be found in the PhD thesis "End-to-End Quality of Service for Multimedia Applications in the Internet" [107].

5.4 Ad hoc Group Meeting Application Prototype

5.4.1 Introduction

The application prototype aims to provide a dynamic real-time multimedia services to mobile users that come together in a physical meeting room with their own mobile devices, and start a collaborative session without any prior human configuration. Users are spontaneously connected with each other, and may share all the resources and services in the room. These resources and services are either provided by the room authority, or brought by users. The ad-hoc group meeting scenarios are set up on the fly with users and services joining and leaving the meeting room in a dynamic fashion. Users' presence is identified automatically by context aware entities, and their devices are dynamically connected and supplied with appropriate tools.

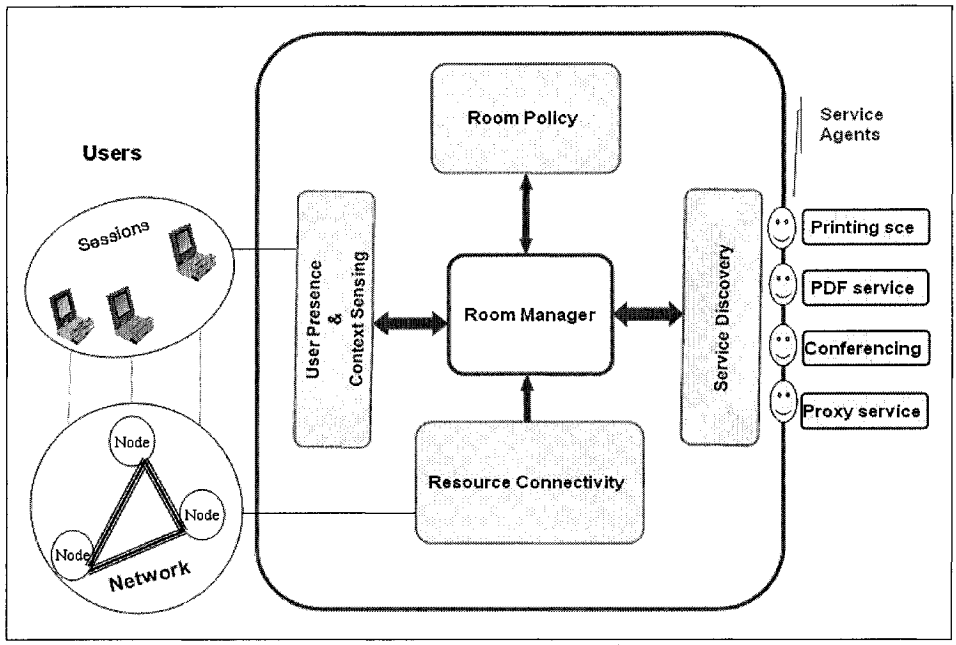

Figure 48. Overview of the ad-hoc group meeting project

The above figure represents the ad-hoc group meeting application prototype with the *room manager* as the central controller. The main purpose of this project is to bring various types of users and services together into a network where they can collaborate with one another and share services in the network.

5.4.2 The Scenario Description

Figure 49 describes a scenario where a user carrying a physical tag, will be detected and recognized by the context awareness entity while entering the room. The Policy Generator is then notified and therefore generates policies that characterize the user.

The room assistant, which is the site assistant in our framework architecture, will then register the current location of the user with SIP proxy server and informs the user personal assistant. The personal assistant informs the room assistant with the role of the user, her/his services and specific policies. The room assistant informs the network serivce agent to dynamically connect the user in case he carries a personal device (a laptop or a PDA), informs the *Service Discovery* and subscribed agents. The service discovery then provides the user with authorized service agents including the conferencing agent which invites him to join the ongoing main collaborative session.

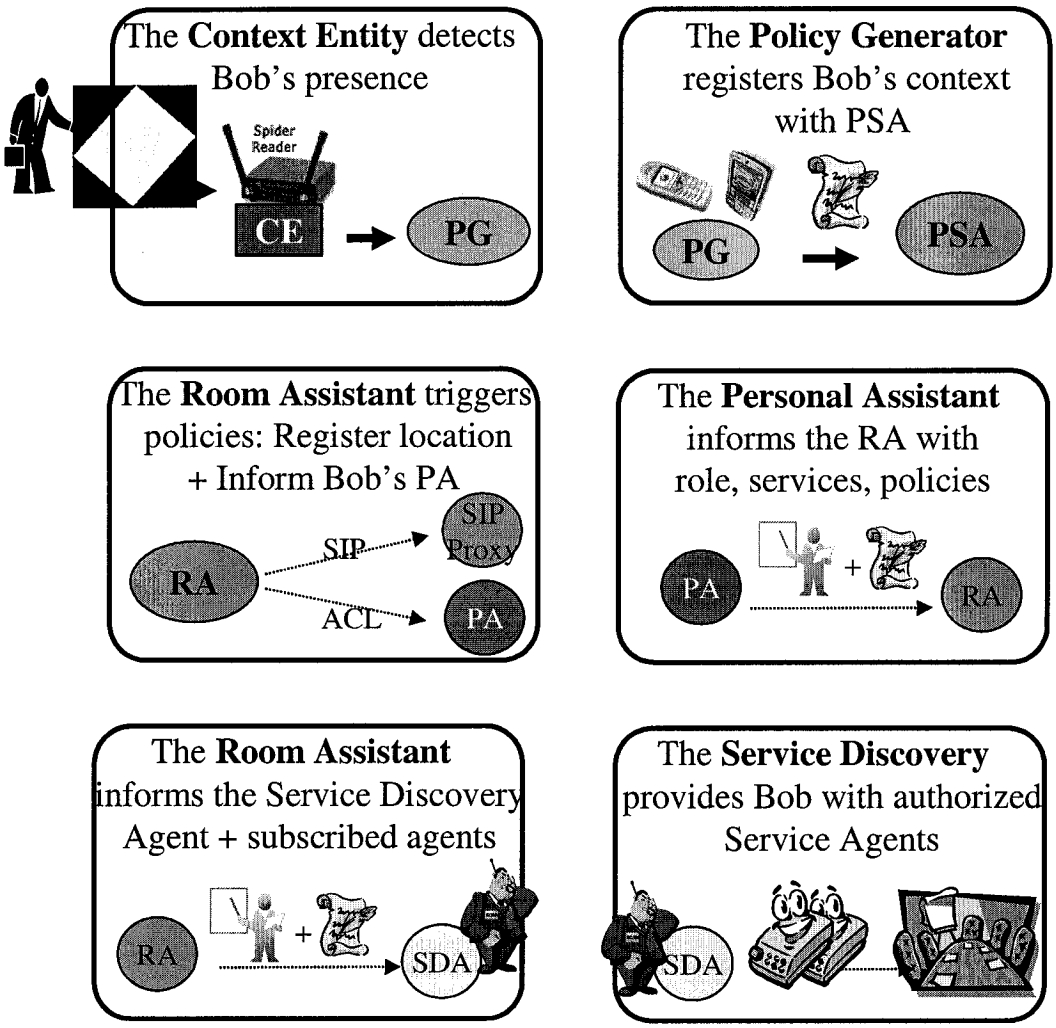

Figure 49: Ad-hoc group meeting scenario

In addition to local resources, users may grant access to resources and services available on their own devices to other users. Such access is performed in accordance with the user's policies as well as the room overall policies, monitored by the policy service agent. The service discovery agent maintains a list of available services and communicates with nearby rooms' service discovery agents to advertise discovered services. Granting access to users services is performed by subscribing these services with the service discovery, so it could advertise them within the room and possibly nearby rooms.

5.4.3 Context-awareness

The presence of users and services is identified spontaneously and agents representing users and services are dynamically activated. User agents and service agents cooperate together to help configuring, setting up and migrating to users' devices when necessary.

The presence has been obtained through radio frequency sensors by associating a tag with each physical entity. A tag that has a UID (Unique IDentification number) is assigned to each entity (user, service, resource etc...) to identify its properties. This UID or TagID is mapped to the entity's profile to obtain its description. An example of such profile is given in figure 15, subsection 3.2.1.

The identification of the tag in the room corresponds to the presence or absence of the entity associated with the tag. The entity is considered to be in the room space as long as the sensing system reads the associated tag.

Mantis Kit from <www.rfcode.com> has been used as a context system for tag sensing. The error in calculating the distance of a tag is approximately ±5 feet, which affects the precision of determining if the tag is inside or outside the room. Therefore, a time limit of 20 seconds between the emissions of beacon has been setup, so if the beacon is not heard for more than 20 seconds, the tag is considered out of range, and then the absence of the associated entity. For instance, if the tag is associated to a user and the tag is not active for more than 20 seconds, it is assumed that the user has left, and therefore the policy generator triggers a SIP BYE action, so that other users will be informed.

A second type of context caption has been used (soft sensors) to detect the network connectivity of users' devices. Specific agents in the network check for new device connections, inform in consequence the policy generator that triggers a policy action for migrating necessary service agents including the conferencing service agent, so that the corresponding user could participate in the room activities.

Figure 50 shows an example of customized user agent that migrates to Bob's PDA (Compaq iPAQ 3850, 206MHz/ 64Mb in memory). It displays the list of users that participate in the ad-hoc group meeting at CBY-B502 room. The user agent provides also the list of available services within the room or nearby environment so that users can select the desired ones among the activities that are currently occuring.

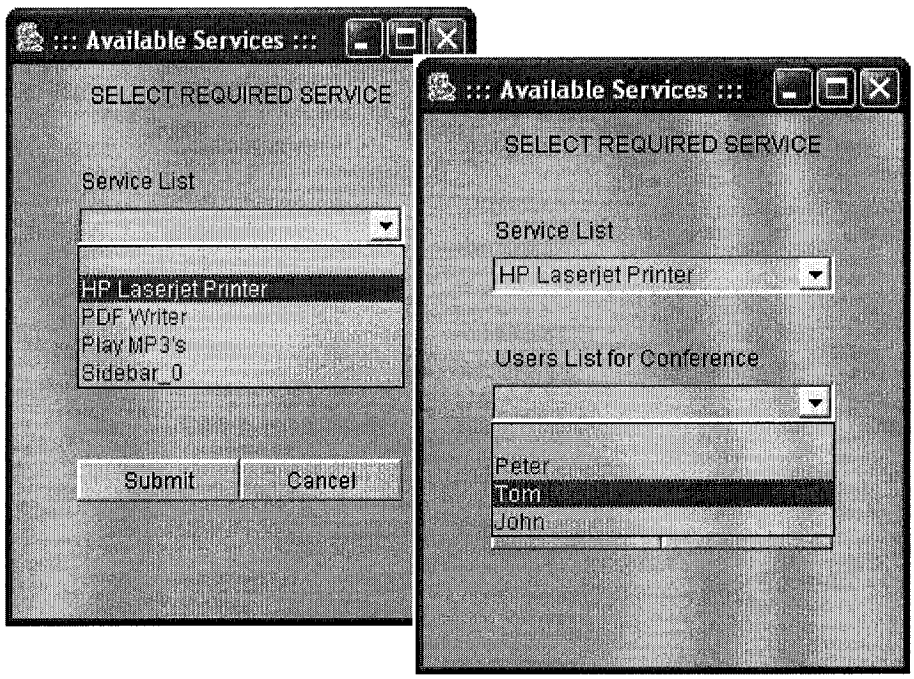

Figure 50. A Snapshot of User Interface on a PDA

In the above example, using his PDA, Bob can use the HP Laserjet Printer (printing service), use the PDF Writer shared by user Tom from his laptop, play MP3's files available monitored by the room assistant, or use the sidebar service which allows Bob to join a conversation amongst a subset of users to which the remaining room participants are not invited.

5.4.4 Discussion

Agents needed for setting up the prototype scenario described above are all provided by the framework, except the service discovery agent that has been implemented as a specific component for this application scenario.

The service discovery agent provides the capability of discovering services and resources to users and applications based on their context. It is a peer-to-peer discovery process where each peer node holds information about services, users, and even context of interest from the peer's point of view. Users that join the group meeting room advertise services and resources they can possibly share with other users in the room with the service discovery agent. The service discovery has teherefore the up-to-date list of all services that are available within the room, and provides then users with athorized services.

The service discovery has the ability to communicate with other service discovery agents in nearby rooms. When a specific service is requested, the service discovery may provide the service agent representing that service from its own list of services or service discovery agents if an agreement is achieved between them.

At least two services are provided within a room entity: the main ad-hoc conferencing service to which users entering the room are invited to participate in, and the sidebar conferencing service wherein subsets of participants involve in private conferencing.

CHAPTER 6

VALIDATION AND EVALUATION

6.1 Introduction

The use of different applications with various scenarios has the goal of validating the implemented framework and to show its feasibility in different situations as claimed in the objective of this research work. The framework has been implemented as a proof of concept. Thus various functionalities of mobility have been demonstrated and the integrity of the framework has been verified. The framework implementation is relying on Java technology. The SIP part uses Tcl / tk, but has been interfaced with a Java wrapper using sockets.

Applications prototypes that have been described in the previous chapter have been successfully implemented using the framework, and they have been successfully demonstrated in many occasions for our partners such as the demos organized at Mitel

Corporation (more than three times) and those organized in the MMARL lab for Mitel and Nortel companies. We have also successfully demonstrated the multimedia service provisioning application prototype in local workshops and forums: Mitel Workshop in 2001 and CITO.

Another aspect of validating the implemented framework was the ability of using the framework with small devices such as PDA's. We made experiment using the service provisioning prototype on IPAQ 3870 PDA with the integration of the multimedia abstraction service described in section 5.2. As shown in figure 48, it was successfully used in our overall scenario. We simulated its browsing capabilities without the authentication and negotiation steps using a wireless 802.11b LAN. The MediABS service agent could move and execute on the PDA which runs MicroFIPAOS agent platform environment. The XML results (as in figure 39, section 5.2.2) are adapted and displayed according to the PDA capabilities (206 MHz Intel StrongARM, 64 MB SDRAM, 16-bit color, and viewable image size – 2.26 in. wide x 3.02 in. tall).

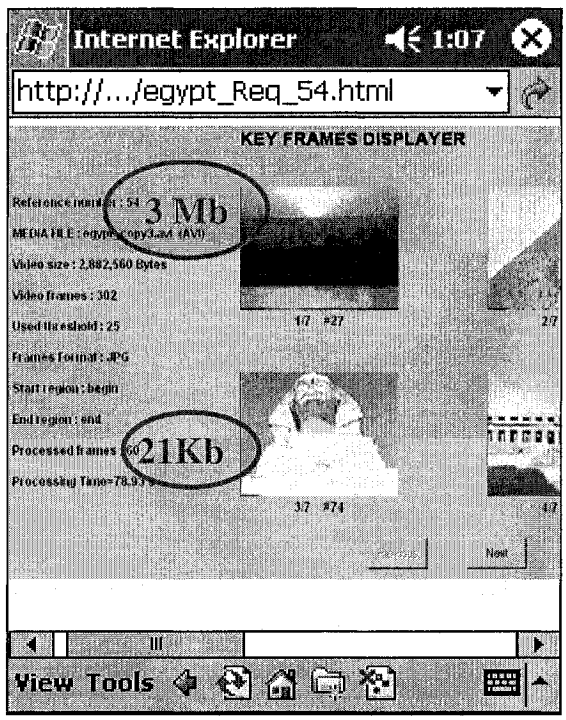

Figure 51. A Sample of the MediABS user interface on PDA.

Figure 51 shows an example of outcome where, out of the entire video mail (about 3MB in total size), the service agent displayed only some representative key frames (21 KB in total size) and a short text report on the iPAQ 3870 PDA.

However, it is also necessary to analyze some performance issues and measure some quantitative parameters including scalability and time responsiveness of the agent-based framework.

Some qualitative and quantitative studies of mobile agents already exist in a variety of mobile computing applications [69][70] that state their appropriateness in such applications. We experimented with the framework on the basis of scenarios similar to those described in the previous chapter and analyze the results depending on each type of agents composing the framework.

Three types of agents could be distinguished in the framework:

- **Service agents** that provide services by performing one or more tasks that require resources;
- **Interface agents** that allow mobile users or other agents to request services;
- **System agents** that establish link between interface agents and service agents.

Service agents advertise themselves to the system agents and interface agents advertise authorized services to users. When a mobile user selects a service, the corresponding service agent migrates to the user's device, and gets the request. Depending on the task to be performed, the service agent may submit the request to the service itself or processes it locally.

We defined the following metrics on which various simulations have been conducted where different types of agents were used:

- The *migration time*: of particular interest for interface agents that migrate from a machine supporting the framework environment to users' devices.
- The *resource consumption*: the usage of CPU and the memory consumption of agents performing their tasks on each user's device, or communicate with the system or service agents.

- The *service agent overload*: when a number of users request the same service agent and overload it.

6.2 Migration time

We measured the migration time of an interface agent over the 10/100 Ethernet LAN and 802.11 wireless LAN. Measurements were done using a simple Java interface agent (17Kbytes) migrating from a Desktop PC supporting V-Team environment to a participant's device. We experimented with three types of devices: a Desktop PC (2.4GHz / 512Mb in memory), a Laptop (1.1 Ghz / 240Mb in memory) with a wireless connection, and a Pocket PC (Compaq iPAQ 3850, 206MHz / 64Mb in memory) with a wireless connection. Table 4 shows a sample of migration time measurement with three types of devices:

Table 6 Agent migration time

	Ethernet (Desktop PC)	802.11 wireless (Laptop)	802.11 Wireless (Pocket PC)
Local agent loading	440 ms	860 ms	2450 ms
Remote agent loading	670 ms	1120 ms	4070 ms
Agent state loading	310 ms	350 ms	1580 ms

The table 6 shows that the average migration time is effected by the network, and also by the Java Class loader mechanism used for moving the agent code (local or remote class loading). But in no case does the migration time exceed 5 seconds. The agent state loading indicates the time necessary to receive an ACL message using FIPA-OS platform.

While a mobile device, undoubtedly more convenient in this environment, the migration time is reduced by 61% when only the agent state is loaded. Depending on the user's context, therefore, some interface agents could migrate in advance to the mobile device, and their updated state would be loaded only when necessary. The advance migration could be triggered by a context policy.

6.3 Resource Consumption

We also examined the CPU usage as the number of agents running on a given end user's device. Our simulation used 1, 3 and 5 simple interface agents migrating to three kinds of devices (Desktop, Laptop and PDA).

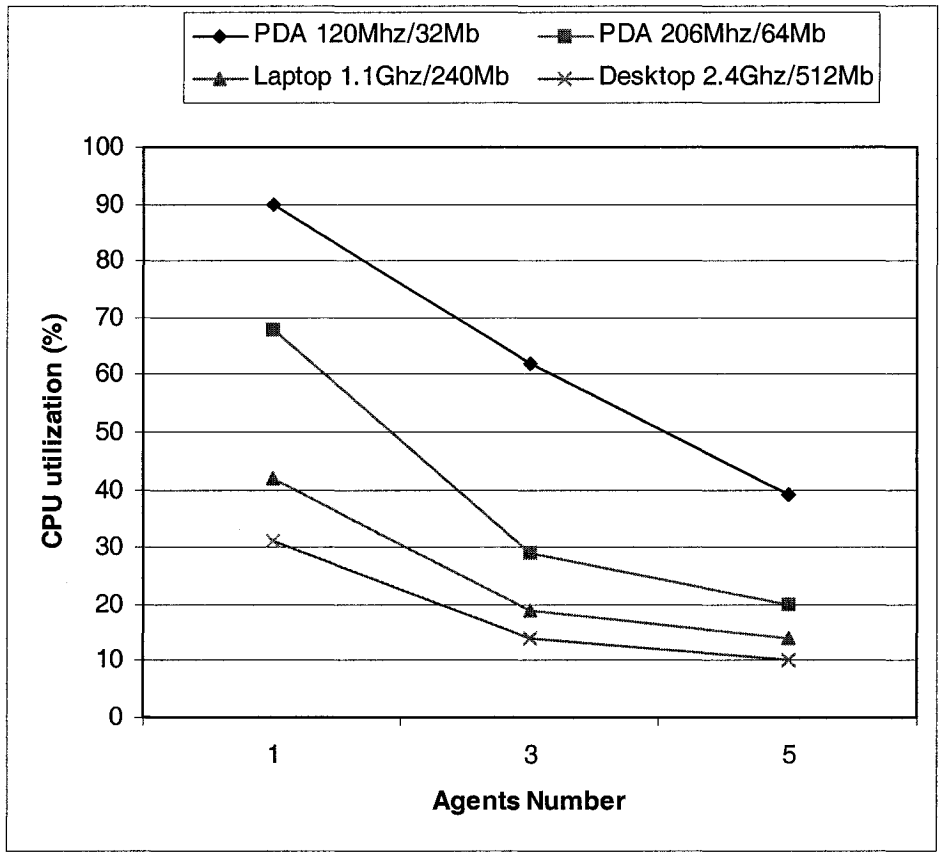

Figure 52. CPU utilization for executing the Framework Agents

Figure 52 shows that, when running alone, an agent consumes between 30 and 70% of the CPU. With five agents running on the same device, a likely case, each agent consumes between 10 and 20%. Depending on the agents' tasks and devices' capabilities, this may affect the overall performance. Devices, especially small ones such as PDA`s, should be equipped with more than 200Mhz and 64Mb memory, so that CPU usage is less than 30% for the execution of each agent.

6.4 Service Agent Overload

We also analyzed the service agent overload, which could happen in the situation where a number of participants request the same service, while the corresponding agent has limited resources to process the tasks required by the multiple requests.

We simulated a collaborative session with 5, 10, 15, and up to 50 participants. In each case, we had all participants request the same service. We identified two kinds of services: (a) an access to a remote service like a shared file, and (b) a service, like an audio conferencing tool, where the agent migrates to each participant's device. In the first case, the service agent is static, and the second case, the service agent is mobile.

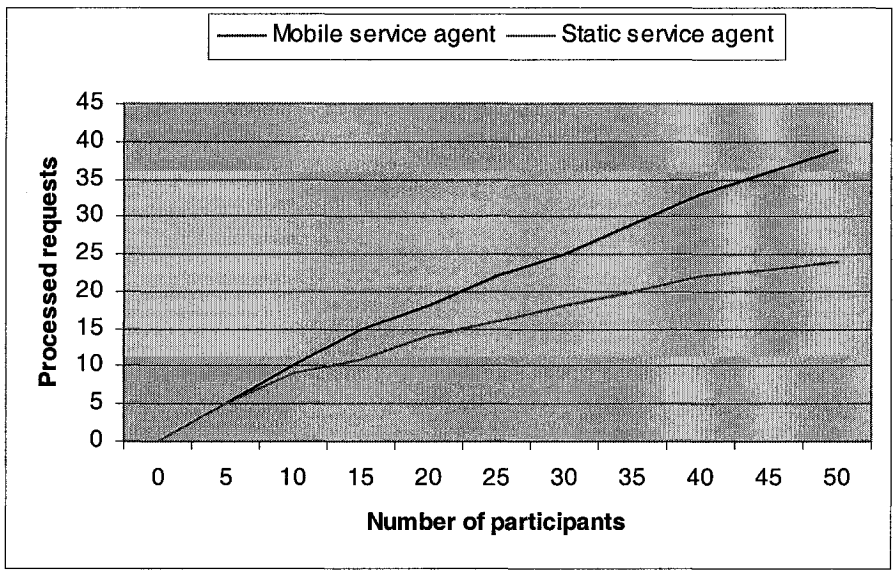

Figure 53. Processed requests using static and mobile service agents

Figure 53 shows that when a smal7l number of participants request the same service, the system performance is the same for static and mobile service agents. But when the number of participants increases, the mobile service agent performs much better. A likely strategy would therefore be to clone agents that represent frequently requested services. Another possibility is that some requests to equivalent services. For instance, if, during an ad-hoc meeting, a number of participants request the printing service, some requests could be delegated to other printers in the meeting room or nearby.

6.5 *Context Effectiveness*

To evaluate the scalability and context effectiveness, we conducted experiments that measure the average delay taken to achieve negotiation of context information as an indication of the system scalability while we present the percentage of time taken to complete a service request-lookup-match-response cycle using our agent framework as an indication of the policy effectiveness in the framework.

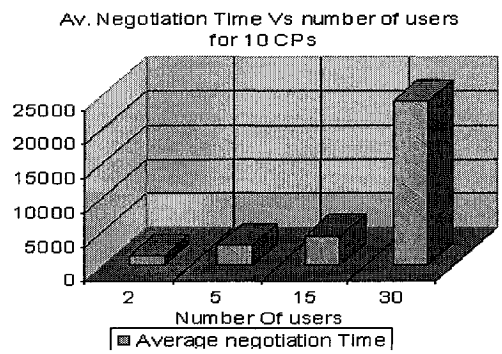

Figure 54.a- Average negotiation time

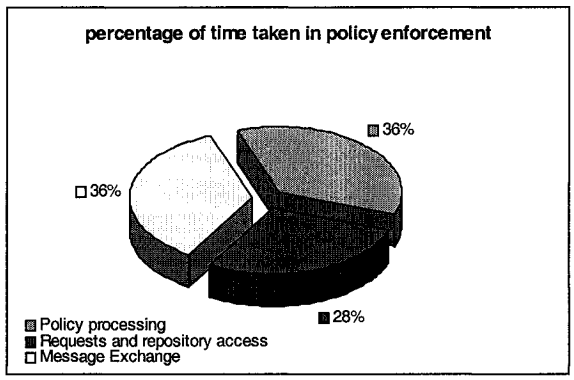

Figure 54-b Percentage of time taken in policy enforcement

Figure 54a shows the average negotiation time taken to reach an agreement, using the context negotiation protocol described in section 4.2, when varying the number of request while fixing the number of context providers. The figure shows that the delay is linearly proportional with the number of request, except for the last value, which is a good indication of the system scalability. The reason for the increase in delay in the last value is attributed to the fact that the number of request is 3 times the number of context providers and thus it took much longer time to settle an agreement between the context providers and the user. However this also indicates that the system may gracefully degrade in performance when it is highly overloaded.

Figure 54b illustrates the percentage of time taken for a complete cycle of service provisioning in case of high overloading. The figure shows that the time taken to process policies is nearly 36% of the total life cycle time while 36% is used for exchanging messages between agents and 28% for repository and information access. This indicates that the overhead of introducing policies to context-aware applications

in extreme overloading is not quite annoying compared to the gain achieved in the framework awareness and the rapid deployment of services for mobile users.

6.6 Summary

This chapter provided an overall evaluation of the framework. The objective of this evaluation was to test various features that distinguish the framework including the use of policies, the network service functionality, the context translation and triggering actions and the use of small devices. A focus was made more on the feasibility and usefulness of the framework by implementing multiple prototypes and scenarios (qualitative analysis) rather than measurements regarding multiple criteria (quantitative analysis). However, measurements have been established for some critical aspects such as the overload that could result from requesting the same service, or the effectiveness of generating context policies and their enforcement.

CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

7.1 Summary of the thesis

This thesis has identified and considered a set of requirements for building a framework to support and simplify the development of mobile services and applications. Mobile and ubiquitous computing have increased the need of mobile users to expect accessing whatever information and services they want, whenever they want and wherever they are. Various computing devices can actually be used in a wide range of situations. Supporting mobility in its different forms and taking into consideration the context clearly help in adapting services and providing them to the

mobile users in a convenient manner. Users do not have to explicitly specify and configure their working environment each time they move from one site to another. Three different prototypes have been developed based on the identified set of requirements composing the framework.

The introductory chapter of this thesis described the emergence and impact of mobility on nomadic users and group of users' services and applications. In more detail, if users are mobile (or communicate with other users that are mobile) then rapid and drastic changes in the underlying quality of communications can occur as a result of users roaming between different sites.

The next chapter (chapter two) surveyed the agent technology currently available to support the development of mobile applications. This chapter highlighted a variety of agent platforms, the policy management, and the use of session initiation protocol (SIP) in managing mobility. The fundamental conclusion of this chapter was that the use of policies at different levels enables agents to adapt their behavior depending on the situation of their use, and also the fact that SIP has features which handles various forms of mobility. Consequently, the integration of SIP to the framework is a key solution to fit with the requirements of developers building services that offer multiple kinds of mobility.

Chapter three examined the concepts and techniques we propose to better help developers building mobile applications by providing a framework that offers common features required in such applications and services. In addition, this chapter described the suitability of using the framework for building context-aware mobile applications. This examination revealed that it is appropriate to translate contextual information to a set of policies, so that services agents representing users' services adapt their behavior and their interfaces according to the overall environment in which mobile users operate.

Then, chapter four explored the actual policy-based context-aware framework components by considering the design issues, and inter-agent interactions. The framework was built to satisfy a set of requirements, obtained from the analysis of our industrial partners' needs and real scenarios.

Chapter five described the implementation of the framework and showed how it supports the development of multiple mobile applications in a convenient and flexible mechanism for dealing with both individual and group oriented services. Three different applications prototypes have been highlighted. The multimedia service provisioning prototype (for individual oriented service) and ad-hoc group meeting prototype (group oriented service) are particularly described in more details.

An evaluation of the framework was detailed in chapter six. The main objective of this evaluation was to test the usefulness of the features and agents provided by the framework (qualitative analysis), and also to establish various measurements regarding multiple criteria (quantitative analysis).

The remainder of this chapter describes the major contributions of the thesis and addresses three areas of potential future work before presenting some concluding remarks.

7.2 Contributions of the thesis

Conducted research has focused on producing an agent-based framework that provides necessary features and services to make it easier for mobile users to interact with computers and the environment, and for designers to build complex mobile applications.

The following points summarize the main contributions of this research work:

- A policy-based agent model that attaches a set of policies to the agent, so that the policy translation and enforcement are tightly bound to the context in which the agent operates.
- A policy generator that gets context information from context-aware entities to produce a set of context policies, so that mobile users could be provided with personalized information and services.

- An agent-based infrastructure for building adaptive applications and services to mobile users. The infrastructure makes use of different basic services and therefore off-loads the need to deal with these services directly by users or applications.
- The use of the framework in setting up various working scenarios from three different applications: Multimedia Service Provisioning, Ad-hoc Group Meetings and Virtual Teams Management.

The following subsections describe each of these contributions in more detail.

7.2.1 The Policy-based Agent Model

This work has proposed a new model of software agent by making policies, their interpretation and their enforcement properties part of the agent. Existing policy-based systems tend to use a centralized entity for monitoring and enforcing policies as described in section 3.1.

The introduction of policies inside the agent provides it with more pro-activeness in facing new situations at visited sites, and increased adaptability while changing from one site to antoher.

We therefore proposed a hierarchical approach in specifying different types of policies. We introduced a policy management entity (PMA) as an application independent service which allows an authority to define and edit policies at a higher level. These policies are then translated to low level policies depending on the application using the policy service agent (PSA) which forwards them to agents composing the application.

7.2.2 Generating Context Policies

This work has proposed to automatically translate effective contextual information to a set of policies. Generated policies are represented, monitored and used by mobile applications in the same manner such as policies edited manually by an administrator or any authorized user.

Managing context at the framework level clearly facilitates the development of context-aware applications. We assume that each site has its own context management system which is responsible of context acquisition, modeling and interpretation. We have introduced the policy generator, an agent that subscribes to the context-aware system for getting appropriate contextual information to be used and delivered to mobile users.

7.2.3 Providing Various Basic Services

This work has proposed the integration of various services at the network level, so that the network will not be used only for transport, but also provides multiple services that relieve users of the constant need to be aware of the details of their computing environment, thereby allowing them to focus on the real tasks at hand.

An example of such services that has been described in section is the integration of the session initiation protocol (SIP) which dynamically locates users and the devices in use. We have also proposed that according to local policies, SIP calls to a mobile user at a visited site could be intercepted by the visited site assistant, which decides to either, reject, redirect or forward them to the user. This is done by re-registering the user with the SIP proxy server once the mobile user is hosted at the visited site (e.g. when a user enters a meeting room, all SIP calls will transit via the entity managing the room meeting).

7.2.4 Applications Prototypes Implementation

This work has proposed the design and the implementation of prototypes of three different applications to demonstrate that the framework architecture can support a number of different applications that make use of mobility and context-awareness.

The use of multimedia service provisioning prototype and ad-hoc group communication prototype were particularly examples of applications for which the framework has been successfully reused and adapted by mainly updating policies and user interfaces. The implementation of the virtual team prototype was mainly useful in demonstrating the use of policies at different levels from the network level to the application level where a user has been assigned a specific role within the team.

7.3 Future Directions

The area of context-aware mobile computing is a broad domain and still undeveloped. This thesis has explored some of the concepts necessary to practically support the development of mobile applications, but it is envisaged that some aspects need to be investigated as future work:

7.3.1 The Framework Context-awareness

We assumed in this thesis that the context-awareness entity provides the framework components with desired contextual information via the policy generator. To determine which kind of information and when it should be provided, a negotiation process using the CLNP (context level negotiation protocol) should be established and therefore common context ontology is required.

The work we have done on the policy generator needs to be enhanced with:

- Defining a specific ontology for mobility context awareness based on the Ontology Web Language (OWL) [103].
- Negotiating the Quality of Context (QoC) to be provided to mobile users at visited locations based on CLNP.
- Defining a set of meta-policies that allow the policy generator triggers appropriate actions for generating new policies according to the received contextual information.

7.3.2 PAN Mobility

This thesis has considered the development of a framework for supporting mobility and especially users' mobility. With the availability of portable computing devices, mobile users are usually equipped with various devices (i.e. laptop, PDA, Phone, etc...). Devices carried out by the user may establish a sort of mobile network (the Personal Area Network) that moves with the user. A key research challenge would be a seamless mobility where the PAN composes with the network at the visited location in transparent manner from the mobile user point of view and therefore exploits efficiently multiple authorized services and resources and may offer his own resources and services (carried within the PAN). At the time the user leaves the visited location, the PAN decomposes smoothly and may compose with the next domain network or a wide range network such as GPRS. This would lead to a new concept of mobility which is the *Network Mobility*.

7.3.3 User Mobility Prediction

As we mentioned in section 6.1, migrating agents in advance to the next destination of a mobile user improves the overall performance and allows the user to be well supported while roaming. This work could be extended by adding the user mobility prediction. This challenging research direction has the objective of defining innovative mechanisms and concepts to accurately determine the next user destination taking into account his profile, the current context, current activities, scheduling tasks, PA instructions and stored data archives to name but few.

7.4 Concluding Remarks

The key conclusion of this thesis is that a new class of mobile applications has arisen with the emergence of mobility with its various forms, breaking many of lacks in the existing tools and applications which are not designed or capable of supporting unique characteristics of mobility.

In a mobile environment, new services are required to provide the mobile application developer with sufficient flexibility to manage the constraints imposed by mobile

communication environments, including the network heterogeneity, the limited device capabilities and the disconnected nature of mobility. This thesis has examined the issues associated with supporting mobility and therefore mobile users and their provisioning with adaptive and personalized services.

This thesis has presented a set of policy-based software agents that attempts to provide developers with a suitable framework for building mobile applications and services. In particular, the framework enables the developer to specify policies that manage dynamically the overall behavior of a mobile application. In addition, the framework enables a developed application to benefit from the context that is automatically translated to policies and integrated to the agents composing the application. This should enable developers to build mobile applications that provide mobile users with more awareness and adaptation while moving from one site to another.

PUBLICATIONS RESULTING FROM THIS THESIS

Refereed Journal Papers

1. Hamid Harroud, Ahmed Karmouch, "*Adaptive Group-based Framework for the Management of Virtual Teams,*" to appear in the Multimedia Tools and Applications Journal.
2. Mohamed Ahmed, Hamid Harroud, Roger Impey, Ahmed Karmouch, "*Agent-based Multimedia Presentation and Adaptation Service,*" the Multimedia Tools and Applications Journal, June 2005.
3. Hamid Harroud, Mohamed Ahmed, Ahmed Karmouch, "*Policy-driven Personalized Multimedia Services for Mobile Users,*" IEEE Transactions on Mobile computing, Vol.2, N° 1, October-March 2003.
4. H. Harroud, A. Karmouch, "*Quality of Service Agent-based Negotiation in Mobile Environments,*" ICON, International Journal on Interoperable Communication Networks, Baltzer Science Publishers, Vol.2, N° 4, 2001.

Refereed Conference Papers

1. H. Harroud, A. Karmouch, "*A Policy Based Context-aware Agent Framework to Support Users Mobility,*" IEEE/AICT July 17-21 2005, Lisbon, Portugal.
2. H. Harroud, M. Khedr, A. Karmouch, "*Building Policy-based Context Aware Applications for Mobile Environments,*" IEEE/IFIP Mobility Aware Technologies & Applications, Oct. 20-22, 2004, Brazil.
3. H. Harroud, A. Karmouch, "*Policy-based Agent Framework for the Management of Virtual Teams,*" IEEE ISSPIT '02, December 18-21, 2002, Marrakech, Morocco

4. M. Ahmed, R. Impey, H. Harroud, A. Karmouch *"Designing Multimedia Service Agents for Mobile users,"* IEEE/IFIP MATA'02, LCNS 2521, Oct. 21,24, Barcelona, Spain

5. H. Harroud, A. Karmouch, *"Context Customization of Virtual Teams Multimedia Collaborative Services,"* Proc. of IMSA 02, August 12-14, 2002 Kauai, Hawaii, USA.

6. Harroud, A. Karmouch, *"Setting Virtual Teams Collaborative Services Using SIP,"* Proc. of the 21th Biennial Symposium on Communications, June 2002, Kingston, Ontario, Canada.

7. H. Harroud, M. Lakhdissi, A. Karmouch, C. Grossner, *"Policy-based Virtual Teams Management,"* IEEE IFIP MMNS'01, October 29-Nov. 1, 2001, Chicago IL, USA

8. H. Harroud, M. Ahmed, A. Karmouch, T. Gray, R. Impey *"Agent-based Personalized Services in A Mobile Computing Environment,"* IEEE/PACRIM'01, August 26-28, 2001, Victoria, B.C., Canada

9. M. Lakhdissi, H. Harroud, A. Karmouch, C. Grossner *"A Policy Management System for Mobile Agent-based Services,"* IEEE/ACM MATA'01, August 14-16, 2001, Montreal, Canada

10. H. Harroud, A. Karmouch, *"Agent-based Personalized Services for Mobile Users over a VPN,"* IEEE/AAA ICEIS'01, July 7-10, 2001, Setubal, Portugal.

11. Harroud H., Ahmed M. and Karmouch A. *"An Agent-based Service Provisioning System for Mobile Users,"* International Symposium on Image/Video Communications over Fixed and Mobile Networks, ISIVC'2000, Rabat Morocco, April 17 –20, 2000

12. H. Harroud, A. Karmouch, T. Gray, S. Mankovski, *"An Agent-based Architecture for Inter-Sites Personal Mobility Management System,"* First International Workshop on Mobile Agents for Telecommunication Applications, October 6-8, 1999, Ottawa, Canada.

References

1. T. Magedanz and A. Karmouch (Guest Editors), "*Mobile Software Agents for Telecommunication Applications*," Journal of Computer Communication, Vol. 23, Issue. 8, 2000.
2. Schulzrinne, H., and Wedlund, E. "*Application Layer Mobility using SIP*," ACM Mobile Computing and Communications Review, vol. 4, no.3, July 2000, pp. 47-57.
3. European Telecommunications Standards Institute, "*Universal Mobile Telecommunication Systems (UMTS)*," http://www.etsi.org/umts/.
4. International Telecommunication Union, "*ITU-T IMT-2000 Activities*," http://www.itu.int/ITU-T/imt-2000/
5. White paper "*Controlling and Reducing the Rising Cost of Enterprise Mobility and Remote Access,*" An Exclusive Report for Enterprise Business and Technology Executives from GRIC Communications, Inc.
6. P. Nixon, V. Cahill, "*Mobile Computing: Technologies for a Disconnected Society,*" IEEE Internet Computing, January-February 1998, pp. 19-21.
7. Cameleon Consortium, "*the European ACTS Research Project (AC341),*" Available:http://www.dfv.rwth-aachen.de/project/cameleon/cameleon.html, 1999.
8. P. Bellavista, A. Corradi and C. Stefanelli, "Mobile *Agent Middleware for Mobile Computing,*" IEEE Computer, March 2001, pp. 73-81.
9. D. Kotz et al., "*Agent TCL: Targeting the Needs of Mobile Computers,*" Internet Computing, July/Aug. 1997, pp. 58-67.
10. E. Kovacs, K. Rohrle, and M. Reich, "*Integrating Mobile Agents into the Mobile Middleware,*" Proc. Mobile Agents Int'l Workshop, Springer-Verlag, Berlin, 1998, pp. 124-135.

11. S. Lippers and A. Park, "*An Agent-Based Middleware: A solution for Terminal and User Mobility,*" Computer Networks, Sept. 1999, pp. 2053-2062.

12. IETF Policy Framework Working Group. "*Policy Terminology Internet Draft,*" March 2001. Available: http://www.ietf.org/internet-drafts/draft-ietf-policy-terminology-02.txt

13. Lucent Technologies. "*CajunView Enterprise Network Management System,*" position paper 2000

14. Cisco Systems, CiscoAssure Policy Networking "*Enabling Business Applications through Intelligent Networking,*" Statement of Direction July 2000

15. S. Franklin, A. Graesser, "*Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents,*" In Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages. Springer-Verlag, 1996. On-Line: http://www.msci.memphis.edu/~franklin /AgentProg.html

16. P. Maes, Ed., "*Designing Autonomous Agents,*" MIT Press, Cambridge, MA, 1990.

17. P. Maes, "*Agents that Reduce Work and Information Overload,*" in Communications of the ACM, 37(7), pp. 31-40, 1994.

18. "*The Agent Society,*" Home page, available at http://www.agent.org/.

19. J. Bradshaw, "*An Introduction to Software Agents,*" in Software Agents, pp.3-46, J. Bradshaw (ed.), MIT Press, Cambridge, 1997

20. B.Falchuk, A.Karmouch, "*A Mobile Agent Prototype for Autonomous Multimedia Information Access, Interaction and Retrieval,*" Proceedings of Multimedia Modeling '97, pp.33-48, Singapore, November 1997

21. V. Pham and A. Karmouch, "*Mobile Software Agents: An Overview,*" IEEE Communications, Special issue on mobile agents and telecommunications, Vol. 36, No. 7, 1998, pp. 26-36.

22. Oracle, "*Oracle Mobile Agents*," April 1997. On-line: http://www.oracle.com/products/networks /mobile_agents /html/ whitepaper.html

23. C. Schramm, A. Bieszczad, B. Pagurek, "*Application-Oriented Network Modeling with Mobile Agents*," to be presented at IEEE/IFIP Network Operations and Management Symposium NOMS' 98, New Orleans, Louisiana, February 1998.

24. P. Buckle and R. Hadingham, "*FIPA and the Internet Revolution*," Key notes speech at the 2nd International ACTS Workshop, Singapore, 1999.

25. "*FIPA Domains and Policies Specification*," Foundation for Intelligent Physical Agents, February 2001. Available: http://www.fipa.org/specs/fipa00089/

26. D. Milojicic, M. Breugst, I. Busse, J. Campbell, S. Covaci, B. Friedman, K. Kosaka, D. Lange, K. Ono, M. Oshima, C. Tham, S. Virdhagriswaran, and J. White, "MASIF*: The OMG mobile agent system interoperability facility*," In Proceedings of Mobile Agents, pages 50--67, Stuttgart, Germany, Sept. 1998.

27. P. Buckle, "*FIPA and FIPA-OS Overview*," Holonic Manufacturing Systems and FIPA Workshop, London, September, 2000

28. "*FACTS: FIPA Agent Communication Technologies and Services*," available at: http://more.btexact.com/profsoc/facts/

29. Cameleon Consortium, "*the European ACTS Research Project (AC341)*," Available:http://www.dfv.rwth-aachen.de/project/cameleon/cameleon.html, 1999.

30. The CASBAH Project - http://www.casbah.org/

31. CRUMPET Project - http://www.ist- crumpet.org

32. Grasshopper – the agent platform http://www.grasshopper.de/

33. M. Zaid "*Personal mobility in PCS*," IEEE Personal Communications, Fourth Quarter 1994.

34. JADE http://jade.cselt.it/

35. Zeus – BT Intelligent Agent Research http://www.opensource.org/

36. Aglet community http://aglets.sourceforge.net/

37. G. Glass, "*Overview of Voyager: ObjectSpace's Product Family for State-of-the-Art Distributed Computing*," 1999.

38. MicroFIPA-OS - http://www. cs.Helsinki.FI/group/crumpet/mfos/

39. F. Bergenti and A. Poggi, "*LEAP: a FIPA Platform for Handheld and Mobile Devices*," ATAL, 2001.

40. J. R. Searle, "*Speech Acts*," Cambridge University Press Cambridge, UK, 1969.

41. FIPA, "*FIPA ACL Message Structure Specification*," FIPA specification no. 61, available at http://fipa.org/specs/fipa00061

42. G. Hughes, and M. Creswell, "*An Introduction to Modal Logic*," Metheun, London, 1968.

43. M.N. Huhns, "*Agent Teams: Building and Implementing Software*," IEEE Internet Computing, February 2000.

44. S. Poslad, M. Calisti, "*Towards improved trust and security in FIPA agent platforms*," Autonomous Agents, Barcelona, Spain, June 2000.

45. M. Sloman, "*Policy Driven Management for Distributed Systems*," Plenum Press Journal of Network and Systems Management, vol 2, no. 4, Dec. 1994, pp. 333-360

46. N. Damianou, N. Dulay, E. Lupu, M. Sloman, "*Ponder: A Language for Specifying Security and Management Policies for Distributed Systems*," Imperial College Research Report DoC 2001, Oct. 2000

47. E. Lupu and M. Sloman, "*Conflict Analysis for Management Policies*," Fifth IFIP/IEEE International Symposium on Integrated Network Management IM'97, San-Diego, May 1997, Chapman & Hall Publishers, pp 430-443

48. M. Sloman. & J.D. Moffett, "*Domain Management for Distributed Systems*," In Proc of the IFIP Symposium on Integrated Network Management, May 1989 pp 505-516.

49. J. Moffett, M. Sloman, "*Policy Hierarchies for Distributed Systems Management*," IEEE Journal on Selected Areas in Communications, Vol. 11 No. 9, Dec. 1993, pp. 1404-1414

50. R. Wies, "*Policies in Network and Systems Management - Formal Definition and Architecture*," Journal of Networks and Systems Management, Vol. 2, No. 1, March 1994, pp. 63-83

51. Edwards, W. K. "*Policies and Roles in Collaborative Applications*," In Proc. ACM CSCW'96, Nov.16-20, Boston, MA, USA, pp. 11-20.

52. J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry "*The Common Open Policy Service (COPS) Protocol*," RFC 2748, January 2000.

53. N. Minsky, V. Ungurcanu, "*A Mechanism for Establishing Policies for Electronic Commerce*," In the 18th International Conference on Distributed Computing Systems (ICDCS). May 1998.

54. M. Amer, A. Karmouch, T. Gray and S. Mankovski, "*An Agent Model for the Resolution of Feature Conflicts in Telephony*," Journal of Network and Systems Management, to be published.

55. V. Sacramento, M. Endler and al., "*An Architecture supporting the development of Collaborative Applications for Mobile Users*," In Proc. of the 13th IEEE International WET ICE, 2004.

56. M. Caporuscio and P. Inverardi, "*Yet another framework for supporting mobile and collaborative work*," In Proc. of the International Workshop on Distributed & Mobile Collaboration (DMC), at the WETICE, Linz, Austria, June 2003.

57. R. Litiu and A. Prakash, "*Developing adaptive groupware applications using a mobile component framework*," Proceedings of the ACM conference on Computer Supported Cooperative Work, Philadelphia, Pennsylvania, USA, 2000.

58. D. Buszko, W.-H. Lee, and A. Helal, "*Decentralized ad hoc groupware API and framework for mobile collaboration*," In Proc. of the International ACM SIGGROUP Conference on Supporting Group Work, Boulder, USA, Oct. 2001.

59. J. Lipnack and J. Stamps, "*Virtual Teams, Reaching Across Space, Time and Organizations with Technology*," John Wiley & Sons 1997.

60. E. Miller, "*An Introduction to RDF*," Online Computer Library Centre, Inc., Dublin, Ohio, May 1998, available: http://www.dlib.org/dlib/may98 /miller/05miller.html.

61. W3 Consortium, "*Composite Capability/Preference Profiles (CC/PP)*," http://www.w3.org/TR/NOTE-CCPP/.

62. M. Lakhdissi, H. Harroud, A. Karmouch, C. Grossner "A *Policy Management System for Mobile Agent-based Services*," IEEE/ACM MATA'01, August 14-16, 2001, Montreal, Canada.

63. B.N. Schilit, M. Theimer "*Disseminating active map information to mobile hosts*," IEEE Network 8(5), pp. 22-32. September-October 1994.

64. B.N. Schilit, N.I. Adams, R. Want, "*Context-Aware Computing Applications*," Proceedings of the Workshop on Mobile Computing Systems and Applications, IEEE Computer Society, Santa Cruz, CA, pp. 85-90, 1994.

65. M. Khedr, A. Karmouch, R. Liscano, T. Gray, "*Agent-based Context Aware Ad hoc Communication*," MATA'02, October 2002.

66. H.Schulzrinne, J.Rosenberg, "*Session Initiation Protocol (SIP) caller preferences and callee capabilities*," Internet Draft, Internet Engineering Task Force, November 2002.

67. J. Rosenberg, "*A Presence Event Package for the Session Initiation Protocol (SIP)*," Internet Draft, Internet Engineering Task Force, January 2003.

68. M. Ahmed and A. Karmouch, "*Video Indexing Using a High-Performance and Low-Computation Color-Based Opportunistic Technique*," Journal of SPIE Optical Engineering, Volume 41, Issue 2, Feb. 2002, pp. 505-517.

69. D. Johansen, "*Mobile Agent Applicability*," Proceedings of the Mobile agents, Springer-Verlag, Berlin, vol. 1477, pp. 9-11. 1998.

70. P. Dasgupta, N. Narasimhan, L.E. Moser and P.M. Melliar-Smith, "*MAgNET: Mobile Agents for Networked Electronic Trading*," IEEE Transactions on Knowledge and Data Engineering, Special Issue on Web Applications, July-August 1999.

71. A.P. Kosoresow, G.E. Kaiser, "*Using Agents to Enable Collaborative Work*," IEEE Internet Computing, Vol. 2, No. 4, July 1998.

72. N. Samaan and A. Karmouch, "*A Mobility Prediction Scheme based on Dempster-Shafer Theory of Evidence*," IEEE Transactions on Mobile Computing, to appear.

73. Brumitt, B., Meyers, B., Krumm, J., Kern, A., and Shafer, S, "*EasyLiving: Technologies for Intelligent Environments*," Handheld and Ubiquitous Computing, September 2000.

74. Stanford, V, "*Pervasive computing goes to work: interfacing to the enterprise*," Pervasive Computing, IEEE, Vol. 1, No. 3, July-Sept. 2002, pp.6 – 12.

75. Steve Vinoski's, "*New Features for CORBA 3.0*," Communications of the ACM, October, 1998.

76. Manuel Román, at al, "*Gaia: A Middleware Infrastructure to Enable Active Spaces*," IEEE Pervasive Computing, Vol. 1, No. 4, Oct-Dec 2002, pp. 74-83.

77. Manuel Roman and Roy H. Campbell, "*Providing Middleware Support for Active Space Applications*," In Proceedings of International Middleware Conference, Rio de Janeiro, Brazil, 2003, pp. 433-454.

78. Manuel Roman, Brian Ziebart, and Roy Campbell, "*Dynamic Application Composition: Customizing the Behavior of an Active Space*," IEEE International Conference on Pervasive Computing and Communications, Dallas-Fort Worth, Texas, March 2003, pp. 169-176.

79. Christopher K. Hess, Francisco Ballesteros, Roy Campbell, and M. Dennis Mickunas, "*An Adaptable Data Object Service for Pervasive Computing Environments*," Proceedings of the 6th USENIX Conference on Object-Oriented Technologies and Systems, San Antonio, Texas, February, 2001.

80. Christopher K. Hess and Roy H. Campbell, "*A Context-Aware Data Management System for Ubiquitous Computing Applications*," International Conference of Distributed Computing Systems, Providence, Rhode Island, May 19-22, 2003.

81. Guanling Chen and David Kotz, "*Context-sensitive resource discovery*," First IEEE International Conference on Pervasive Computing and Communications, Fort Worth, TX, March 2003, pp. 243-252.

82. Guanling Chen and David Kotz, "*Solar: An open platform for context-aware mobile applications*," First International Conference on Pervasive Computing, Zurich, Switzerland, August 2002

83. Guanling Chen and David Kotz, "*Context aggregation and dissemination in ubiquitous computing systems*," Fourth IEEE Workshop on Mobile Computing Systems and

84. Guanling Chen and David Kotz, "*Solar: Towards a flexible and scalable data-fusion infrastructure for ubiquitous computing*," Workshop on Application Models and Programming Tools for Ubiquitous Computing at Third International Conference on Ubiquitous Computing, Atlanta, GA, October 2001

85. S. S. Yau, F. Karim, et al, "*Reconfigurable Context-Sensitive Middleware for Pervasive Computing*," IEEE Pervasive Computing, Vol. 1, No. 3, July-September 2002, pp. 33-40

86. S. S. Yau, F. Karim, "*Context-Sensitive Middleware for Real-time Software in Ubiquitous Computing Environments*," IEEE Int'l Symp. on Object-Oriented Real-time Distributed Computing (ISORC 2001), May 2-4, 2001, Magdeburg, Germany, pp. 163-170.

87. T. Kanter, "Attaching *Context-Aware Services to Moving Locations*," IEEE Internet Computing, March-April 2003, Vol. 7, No. 2, pp. 43-51.

88. T. Kanter, "*Going Wireless: Enabling an Adaptive and Extensible Environment*," Special issue on Personal Environment Mobility in Multi-Provider and Multi-

Segment Networks of the ACM Journal on Mobile Networks and Applications (MONET), Vol.8, No. 1, February 2003, pp.37-50.

89. Hong J.I, and J.A. Landay, "*An Infrastructure Approach to Context-Aware Computing*," Human-Computer Interaction (HCI) Journal, pp. 2-3, 2001.

90. Sim K.M., Wang S.Y., "*Flexible Negotiation Agent With Relaxed Decision Rules*," Systems, Man and Cybernetics, Part B, IEEE Transactions on, Vol. 34, No. 3, June 2004, pp.1602-1608.

91. Perich F., Joshi A., Finin T., Yesha Y, "*On data management in pervasive computing environments*," Knowledge and Data Engineering, IEEE Transactions on, Vol. 16, No. 5, May 2004, pp.621 – 633.

92. Lehti P., Fankhauser P., "*XML data integration with OWL: experiences and challenges*," International Symposium on Applications and the Internet, 26-30 Jan. 2004.

93. Phelps S., Tamma V., Wooldridge, M.; Dickinson, I.; "*Toward open negotiation*," Internet Computing, IEEE, Vol. 8, No. 2, March-April 2004, pp.70 – 75.

94. Bellavista P. et al, "*Dynamic binding in mobile applications*," Internet Computing, IEEE, Vol. 7, No. 2, March-April 2003, pp.34 – 42.

95. Ganesan P., Karmouch A. "*Context Awareness in Ad hoc Communications*," IEEE PACRIM, August 2003, pp. 919-922.

96. M. Khedr, A. Karmouch, "*A Semantic Approach for Negotiating Context Information in Context Aware Systems*," IEEE Intelligent Systems Magazine, to appear.

97. A. Karmouch, A. Galis, R. Giaffreda, T. Kanter, et al., "*Context Aware Research Challenges in Ambient Networks*," IEEE/IFIP Mobility Aware Technologies & Applications, Oct. 20-22, 2004, Brazil.

98. Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "*Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)*," BCP 85, RFC 3725, April 2004.

99. FIPA 98 Specification Part 11: "*Agent Management Support for Mobility*," Foundation for Intelligent Physical Agents, October 1998. (Was) available at <http://www.fipa.org/spec/fipa8a27.doc>

100. Vakil, F., Dutta, A., Tauil, M., Baba, S., Nakajima, N., Shobatake, Y. and Schulzrinne, H. "*Supporting Service Mobility with SIP*," IETF, Dec. 2000. {Internet Draft draftitsumo-sip-mobility-service-00}.

101. C. Perkins (Editor), "*IP mobility support for IPv4*," RFC 3220, January 2002.

102. A. Campbell, J. Gomez, C-Y. Wan, S. Kim, Z. Turanyi, and A. Valko, "Cellular IP, work in progress," draft-ietf-mobileip-cellularip-00, January 2000.

103. "*OWL Web Language Ontology Reference*," http://www.w3.org/TR/owl-ref.

104. G. Wallace, "*The JPEG still picture compression standard*," Communications of the ACM Magazine, Volume 34, Issue 4, pp. 30-44, April 1991.

105. Video Edit Magic software, "http://www.deskshare.com/vem.aspx".

106. M. Ahmed and A. Karmouch, "*Video Indexing Using a High-Performance and Low-Computation Color-Based Opportunistic Technique*," SPIE Optical Engineering Journal, Volume 41, Issue 2, pp. 505-517, Feb. 2002.

107. F. M. Sallabi, A. Karmouch, "*End-to-End Quality of Service Support for Multimedia Applications in the Internet*," PhD Thesis, University of Ottawa, SITE, 2001