Universidad Carlos III de Madrid
Escuela Politécnica Superior

Departamento de Informática
Doctorado en Ciencia y Tecnología Informática

Tesis Doctoral

Intelligent Techniques for Context-Aware Systems

Nayat Sánchez Pi

Dirigida por

Javier Carbó Rubiera

José Manuel Molina López

Colmenarejo, 2011

E-mail: nayat.sanchez@uc3m.es

Telephone: +34 91 856 1327

Address:

Grupo de Inteligencia Artificial Aplicada
Departamento de Informática
Universidad Carlos III de Madrid
Av. de la Universidad Carlos III, 22
Colmenarejo 28270 — Spain

Intelligent Techniques for Context-Aware Systems

Autor: Nayat Sánchez Pi
Directores: Javier Carbó Rubiera
José Manuel Molina López

Firma del Tribunal Calificador:

	Nombre y Apellidos	Firma
Presidente: D.	....	....
Vocal: D.	....	....
Vocal: D.	....	....
Vocal: D.	....	....
Secretario: D.	....	....

Calificación: ....

Colmenarejo, ... de .... de 2011.

To my parents Georgina and Manuel

Contents

List of Figures

Resumen

La Inteligencia Ambiental y los entornos inteligentes hacen hincapié en una mayor facilidad de uso, soporte de servicios más eficientes, el apoderamiento de los usuarios, y el apoyo a las interacciones humanas. En esta visión, las personas estarán rodeadas de interfaces inteligentes e intuitivas incrustados en objetos cotidianos que nos rodean y los sistemas desarrollados para este ambiente deberán reconocer y responder a la presencia de individuos de una manera invisible y transparente a ellos. Esta tesis se centra principalmente en el problema abierto en los sistemas sensibles al contexto: la evaluación de estos sistemas en los entornos de Inteligencia Ambiental. Con respecto a este tema, se argumenta que debido a las propiedades altamente dinámica de los entornos de inteligencia ambiental, debería existir una metodología para la evaluación de estos sistemas, teniendo en cuenta el tipo de escenarios. Sin embargo, con el fin de apoyar con una base sólida para la discusión, algunos elementos deben ser discutidos también. En particular, nosotros:

- Usamos una plataforma comercial que nos permite diseñar y gestionar la información contextual de los sistemas sensibles al contexto a través de un gestor de contexto incluido en la arquitectura;
- Analizamos la representación formal de esta información contextual a través de un sistema basado en el conocimiento (SBC);
- Discutimos las posibles metodologías que se utilizarán para el modelado del conocimiento en SBC y nuestra aproximación y propuesta;
- Discutimos las razones del por qué los agentes inteligentes son una técnica válida para ser aplicada a los sistemas en entornos inteligencia ambiental;
- Proponemos un sistema multi-agente (SMA), con una arquitectura genérica que se puede aplicar a una gran clase de aplicaciones de inteligencia ambiental;
- Proponemos una interfaz de usuario multimodales y su integración con nuestro SMA;
- Proponemos una metodología de evaluación de los sistemas sensibles al contexto en los escenarios de inteligencia ambiental.

La formulación de los elementos antes mencionados se hizo necesaria en la medida que esta tesis se ha desarrollado. La falta de una metodología de evaluación de los sistemas sensibles al contexto en entornos de inteligencia ambiental, nos llevó a los objetivos principales de esta tesis. En este sentido, en esta tesis:

- Proporcionamos un estado del arte actualizado y exhaustivo de este asunto;

- Examinamos las propiedades y características de los escenarios de inteligencia ambiental y nuestra aproximación usando técnicas inteligentes;

- Proponemos una metodología de evaluación para este tipo de sistemas y experimentalmente probamos nuestra metodología en diversos escenarios de inteligencia ambiental.

Abstract

Nowadays, with advances in communication technologies, researches are focused in the fields of designing new devices with increasing capabilities, implanting software frameworks or middleware to make these devices interoperable. Building better human interfaces is a challenging task and the adoption of Artificial Intelligence (AI) techniques to the process help associating semantic meaning to devices which makes possible the gesture recognition and voice recognition.

This thesis is mainly concerned with the open problem in context-aware systems: the evaluation of these systems in Ambient Intelligence (AmI) environments. With regard to this issue, we argue that due to highly dynamic properties of the AmI environments, it should exist a methodology for evaluating these systems taking into account the type of scenarios. However in order to support with a solid ground for that discussion, some elements are to be discussed as well. In particular, we:

- use a commercial platform that allows us to design and manage the contextual information of context- aware systems by means of a context manager included in the architecture;
- analyze the formal representation of this contextual information by means of a knowledge based system (KBS);
- discuss the possible methodologies to be used for modelling knowledge in KBS and our approach;
- give reasons why intelligent agents is a valid technique to be applied to systems in AmI environments;
- propose a generic multi-agent system (MAS) architecture that can be applied to a large class of envisaged AmI applications;
- propose a multimodal user interface and its integration with our MAS;
- propose an evaluation methodology for context-aware systems in AmI scenarios.

The formulation of the above mentioned elements became necessary as this thesis was developed. The lack of an evaluation methodology for context-aware systems in AmI environments, where so many issues to be covered, took us to the main objective of this thesis. In this regard:

- we provide an updated and exhaustive state-of-the-art of this matter;
- examine the properties and characteristics of AmI scenarios and our approach using intelligent technologies ;
- put forward an evaluation methodology and evaluation measures for AmI scenarios.

Agradecimientos

Me gustaría no pasar por alto el momento de agradecer a todas aquellas personas que han hecho este trabajo posible. En primer lugar a mis padres, Georgina y Manuel, a quienes también va dedicado este trabajo, por haberme trazado el camino que de alguna manera me ha traído hasta aquí. A mi abuela Esperanza, por ser la inspiración y alegría en que me miro, y al resto de mi familia: tía Tere, Brendy, Tata y Gusti a quienes recuerdo cada día.

De manera especial a mis directores de tesis: José Manuel Molina, a quién en particular no terminaré de agradecerle porque sin su absoluto apoyo y paciencia, esto carecería de sentido; y Javier Carbó por todo su conocimiento y sabios consejos. ¡Gracias!

A la Universidad Carlos III y a todos mis compañeros del grupo GIAA, los que están y los que ya no: Jesús, Antonio, Miguel Ángel, Juan, David, Gonzalo, Rodri, Álvaro, Kike, Mirren y Mar por todos los momentos.

A Luis, por su amistad, su infinita paciencia, sus horas de sueño y por enseñarme a navegar. Gracias a él, estos últimos momentos han sido más llevaderos.

A Eleni Mangina quién me recibió de manera especial en la University College Dublin durante mi estancia predoctoral y a quién admiro muchísimo. A Luis Correia, quién igualmente fue un anfitrión insuperable en la Universidade de Lisboa.

A mis amigos de siempre, esos de aquí y de allá: Ernesto, Liena, Pepe, Patri, Boris, Lily, Pita, Aimara, Maria Estela, Liset, Salim, Herline, Yaney, Jose, por todos los momentos y su infnito cariño.

A todos, muchas gracias.

Nayat Sánchez Pi
Colmenarejo, septiembre 2011.

1
Introduction

1.1 Motivation

The 21st century is rapidly becoming ubiquitous and these challenges are entering in daily life activities and environments. The Ubiquitous Computing idea envisioned by Weiser in 1991, presents a vision of a world where computers would be embedded in everyday life where people could communicate with computers relating to services in a way where the network infrastructure should be transparent to the user itself (Weiser, 1991).

This computing paradigm has recently evolved to a more general one known as Ambient Intelligence (AmI). Ambient Intelligence and Smart Environments (SmE) emphasize on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. In this vision, people will be surrounded by intelligent and intuitive interfaces embedded in everyday objects around us and an environment recognising and responding to the presence of individuals in an invisible way (Kovács & Kopácsi, 2006).

AmI represents, in other words, a new generation of user-centred computing environments aiming to find new ways to obtain a better integration of the information technology in everyday life devices and activities. Ideally, people in an AmI environment does not notice these devices, but they will benefit from the services they are able to provide. Such devices are aware of the people presence in those environments by reacting to their gestures, actions and context (Aarts et al., 2001). In order to work efficiently, software running on these devices may have some knowledge about the user, it means that there is an increasing need of improve context awareness and knowledge sharing without interfering with user's daily life activities.

Nowadays, with advances in communication technologies, researches are focused in the fields of designing new devices with increasing capabilities, implanting software frameworks or middleware to make these devices interoperable. Building better human interfaces is a challenging task and the adoption of Artificial Intelligence (AI) techniques to the process help associating semantic meaning to devices which makes possible the gesture recognition and voice recognition.

AmI environments are integrated by several autonomous computational devices of modern life ranging from consumer electronics to mobile phones. AmI has several spheres of application like: Transportation, Health, Education, Business, etc, but recently the interest in Ambient

Intelligence Environments has grown considerably due to new challenges posed by society, demanding highly innovative services such as vehicular ad hoc networks (VANET), Ambient Assisted Living (AAL), e-Health, Internet of Things and Home Automation among others. These society challenges force developers to take into account growing demands of users.

Although there are already solutions that have been successfully implemented to deliver the right service to the right user at the right time, there are several important issues not tackled in these kind of approaches: generic architecture,the self-adaptation and the user-centric evaluation paradigm. Due to highly dynamic properties of the above introduced environments, the software system running on them has to face problems such as: user mobility, service failure, resources or goal changes which may happen in any moment. To cope with these problems, such system must senses the environment, and acts on it, over time in pursuit of its own benefit.

That is why there is a need of special kind of software that should combine ubiquity, context-awareness, intelligence and natural interaction in an AmI environment. The system has also to adapt not only to changes in the environment, to be autonomous and self-managed but also to the user requirements and needs. User has to take the relevant role providing an evaluation of the system behaviour while using it or at least once it has been used. One of our goals is to evaluate enhanced user experience in the course of using our system and provide automatic adaptation taking into account changes in user preferences and environment. So, generic user-centred evaluation system provides users with the possibility of having a proactive role when using the system. It is on users hands to provide the system with a feedback of the correctness of the provided services. Users will then be capable of specify the right or wrong context information at a high-level concept so that the system could learns from it and self-adapt its behaviour for future times.

1.2 Objectives

With the appearance of mobile devices and communication technologies, concepts like pervasive computing and ubiquitous (Weiser, 1991) become an important spark which ignited the development of context-aware systems. The state of the art provided in chapter 2, arouses the principal motivation of this thesis. There are important issues in the designing and development of frameworks for AmI environments to be taken into account: architecture, context model, context processing, resource discovery, historical context data. But there main point is the lack of a generic architecture that address all of these problems and also that provides with a multimodal interface and a user-centric evaluation. Based on this motivation, the objectives of this Ph.D thesis will define a research line that is highlighted in the following ideas:

1.2.1 Objective 1: To design and develop a centralized system for AmI environments

First objective is to design and develop a centralized system in order to provide context aware services in AmI environments. The use of a commercial platform with a layered architecture allow us to manage the contextual information by means of a context manager included in the architecture. The formal representation of this contextual information is conceptualized by means of a knowledge based system (KBS). There are also several methodologies for the development of these systems but the most known is Knowledge Acquisition and Document Structure (KADS) and it is the one we will use to represent, in a formal way, our problem defined in the platform. CommonKADS (Schreiber et al., 1999) (which is becoming the de facto European standard successor of KADS) is the knowledge engineering methodology we use.

1.2.2 Objective 2: To develop a distributed system architecture based on intelligent agents

Second objective is to develop a generic architecture using intelligent agents that can be applied for any scenario in AmI. For a large class of the envisaged AmI applications, the added value of new services is likely to be for people in ordinary social contexts. Such applications beg for technologies that are transparent, so that their functional behaviour can be easily understood by users. Intelligent agents seem to be the appropriate solution for AmI environments since they provide autonomy and proactivity. The design and implementation of the system based on agents is maximized using Gaia methodology (Franco Zambonelli & Wooldridge, 2003). Benefits over the centralized solution are evaluated.

1.2.3 Objective 3: To include a multimodal user interface

Third objective consists of including a multimodal user interface in order to ensure natural and intelligent interaction. In AmI it takes an important role because users can naturally interact with the system and thus perceive it as intelligent. To ensure such a natural and intelligent interaction, it is necessary to provide an effective, easy, save and transparent interaction between the user and the system. So, a multimodal dialogue system, is our approach. The scheme used for the development of these systems includes several generic modules that deal with multiple knowledge sources and that must cooperate to satisfy users requirements.

1.2.4 Objective 4: To define a generic evaluation methodology for this systems in AmI environments

Fourth objective deals with the evaluation of the system, not only from the software developer's point of view but also from the user's point of view. Our aim is that the user provide with a feedback so we can collect this information for being used later in the adaptation process. Systems must be autonomous and self-managed. The system adapt not only to changes in the environment but also to the user requirements and needs. In our system, users take the relevant role providing an evaluation of the system behaviour while using it or once it has been used (online/ offline: OnE/ OffE). Adaptation process will come fullcircle and we use an utility-based approach that aims at delivering the best possible decision.

1.3 Outline of thesis proposal

The objectives of this Ph.D thesis are: i) Proposing a centralized system that would use a knowledge based system (KBS) to model, represent and manage the contextual information for AmI environments. ii) Defining the KBS using a platform (Appear) in order to manage the contextual information for practical mobile applications, including a speech based interface. iii) Proposing a distributed system based on intelligent agents in order to provide context-awareness and situation sensitivity. iv) Analysing several domains of application for testing centralized and distributed proposals and evaluate the benefits of a distributed architecture vs centralized one. v) Developing a methodology of evaluation capable of providing feedback based on user experience while using the system. This final proposal will have come full circle encouraging self-adaptation and learning to the context-aware system.

1.4 Document structure

The rest of this document is structured as follows:

- Chapter 2: where the state of the art and theoretical basis of our discussion are lay down; context-aware systems; multi-agent systems; dialogue system and evaluation systems.
- Chapter 3: where the context-aware system development is presented.
- Chapter 4: where the application scenarios are presented.
- Chapter 5: where conclusions and future lines of work are outlined.

2 State of the Art

Nowadays there is a significant transformation in the way people have in mind the concepts of computing and the way they look at their interaction with new technologies. Computational resources are every time closer to all the people and that is possible because of the development of devices such as Personal Digital Assistants and mobile phones. These devices have gained a lot of popularity in these days and are increasingly being networked.

Due to advances in communication technologies such as sensor networks and radio frequency identification (RFID), ubiquitous computing is increasingly entering in every aspects of our life and sectors, opening a world of unprecedented scenarios where users interact with electronic devices embedded in environments that are sensitive to the presence of users (Lyytinen et al., 2004). These systems combine ubiquitous information, communication, with enhanced personalization, natural interaction and intelligence. The use of this context offers the possibility to tailor a new type of advanced applications.

Ambient Intelligence (AmI) paradigm builds upon ubiquitous computing, personalization and human-centric computer interaction design. In (Aarts et al., 2001; Zelkha & Epstein, 1998) AmI is characterized and closely related to the long term vision of an intelligent service system in which technologies are able to automate a platform embedding the required devices for powering context aware, personalized, adaptive and anticipatory services. In that way, for these systems working efficiently and transparently it is necessary to have some knowledge of the user.

In this chapter we provide a background of system design challenges related to ambient intelligence systems. The relationship with other technologies of human-centric computer interaction like context awareness, intelligent agents and multimodal interaction are stated. First, we provide a background of context aware computing in AmI scenarios. Afterwards, intelligent agents technologies is presented as a solution for systems in the scope of ubiquitous computing and ambient intelligence. Multimodal interaction paradigm is later put forward as an intuitive way to access the information in this systems. Finally the evaluation is addressed.

2.1 Context-aware systems

The use of nowadays technology like notebooks, PDAs and smart phones gave birth to the concept of: mobile computing. Mobile computing field had an increasing attention few years ago, as many systems were designed towards this direction and to be context-aware with the aim of optimizing and automating the distribution of their services in the right time and in the right place. There is another concept introduced first by Weiser (Weiser, 1991): pervasive computing, referring to the seamless integration of devices into the users everyday life. It is also named ubiquitous computing, so devices should vanish into the background to focus on the user and his tasks rather than being aware of the computing devices and technical issues.

Context-aware systems appear in computer science as one field in the wide range of pervasive computing. Context-aware systems should be able to adapt their operations to the current context without explicit user intervention and taking environmental context into account. Particularly when it comes to using mobile devices, as context data may change so rapidly, it is desirable that programs and services react specifically to their current location, time and other environment attributes and adapt their behaviour according to the changing circumstances.

We start hearing about context aware systems when in 1994 Want et al. (Want et al., 1992) introduced their Active Badge Location System which is considered to be one of the first context-aware applications. This system was able to determine a user's current location through infrared technology, which was used to forward phone calls to a telephone close to the user. Later came a couple of location aware systems like tourist guides: (Abowd et al., 1997; Burrell & Gay, 2002; Cheverst et al., 2000; Espinoza et al., 2001; Kerer et al., 2004; Priyantha et al., 2000a; Sumi et al., 1998). Context-aware systems dealing with location information are treated in literature apart from other context-aware systems and it is due to the widespread and the demand for the increasing spread of mobile devices. The use of different types of status context such as temperature, noise, light and location allows the combination to high-level context objects. In order to build more adaptive, useful and user-friendly systems, some of the context-aware systems in literature made an approach. This is the case of (Muñoz et al., 2003) which adds context-awareness to support information management within a hospital setting.

Next subsections will discuss some relevant points while dealing with context-aware systems. First of all we give a proper definition of context in terms of computer science. After we state clear the communication technologies and infrastructures involved to serve context-aware systems. Also, the most referred context-aware architectures and the special requirements to provide these context-aware services in AmI scenarios.

2.1.1 Context definition

In Human Computer Interaction (HCI), there are two important concepts defined by (Schmidt, 2002): implicit input that is based on the concept of using human activity in the real world as input to computers; and explicit input that is the interaction where the user needs to explicitly shift from the real environment to the virtual one. The above definitions provides the substantial benefits of thinking beyond shifting from explicit human computer interaction where the user gives explicit input and specify the task to implicit human computer interaction where all of this become transparent and context aware. On the other hand, we have Ambient Intelligence that builds upon the implicit characteristics of HCI. So, context aware paradigm becomes an important issue in the designing of AmI systems. Context aware computing is characterized by systems and devices that can recognize the context information. This context information describes the situation of the different entities involved in the communication process (users, place, environment and computational devices). Contextual information can be either the type of the device exploited to access a service, or the location of the user, or its personal preferences, etc. The computing paradigm in which applications can discover and take advantage of contextual information is the so called context-aware computing.

Context-awareness means that one is able to use context information. A system is context-aware if it can extract, interpret and use context information and adapt its functionality to the current context of use. The challenge for such systems lies in the complexity of capturing, representing and processing contextual data. In human-human interaction (HCI), a great deal of information is conveyed without explicit communication. Context aware computing could be viewed as an extension of implicit input. HCI and context aware computing together limit the need for explicit input since the input data is either captured or derived through context aware computing. Context aware systems can also infer the context and adapt the system in such a way that it can better capture the implicit input.

A large number of definitions of the terms context and context-awareness has been proposed in the area of computer science, since the term context-aware computing was first introduced by Schilit et al. in 1994 (Schilit et al., 1994). Context experienced various characterizations using synonyms such as an application's environment (Hull et al., 1997) or situation (Brown, 1996). Many authors define context by example (Brown et al., 1997; Dey, 2001; Gross & Specht, 2001; Ryan et al., 1998) where Dey presents alternative views on context and its definition which can be categorized: places (rooms, buildings etc.), people (individuals, groups) and things (physical objects, computer components etc.) and enumerate context elements like location (an entity's position, co-location, proximity etc.), identity (each entity has a unique identifier), time (used for timestamps to accurately define situation, ordering events, etc.), status (or activity, meaning the intrinsic properties of an entity, e.g., temperature and lightning for a room, noise, processes running currently on a device etc.) as well as the beliefs, desires, commitments, and intentions of the human (Chen, 2004). But one of the most accurate definition of context is the one given by Dey and Abowd (Dey & Abowd, 2000) where

"any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves."

2.1.2 Sensing context: Communication technologies involved

Once we have classified context information, there is a need of sensing it for being able to be used by systems and applications. The context information needed may be obtained from a variety of sources, such as networks, devices, or by applying sensors and browsing user profiles.

Identity and time are contextual information not difficult to obtain, of course, from the built-in computer's MAC address or clock of the computer. Many earlier applications correlate the location information of each device with timestamp, such as Active Badge (Want et al., 1992), ParcTab (Want et al., 1996), and Cyberguide (Abowd et al., 1997).

Location and position have been widely investigated for their use in context-aware systems, Since the location is an important context that changes whenever the user moves, so a reliable location-tracking system is critical to many context-aware application. There exist typical techniques dealing with the explicit will of the user of supplying his location to the system. But what would happened if the user do not agree to cooperate or even if he forgets to let the system know his positioning. So, this is why it is important to have a location-positioning system in order to provide with an automatic location sensing. There are several approaches for sensing location context for outdoors, indoors or hybrid approaches. When discussing location and position, issues such as co-location and proximity are also relevant (Hightower & Borriello, 2001).

The first indoor system was the Active Badge Location System (Want et al., 1992), later came Active Bat Location System (Harter et al., 2002). The Cricket Location Support System (Priyantha et al., 2000b) is another system like the Active Bat system, using ultrasonic time-of-flight data and a radio frequency control signal. RADAR (Bahl & Padmanabhan, 2000) is a building-wide tracking system based on the IEEE 802.11. MotionStar magnetic tracker (Raab et al., 1979) offers a classic position-tracking method. Easy Living (Orr & Abowd, 2000) provides one example of the using computer vision technology to figure out where things are. In the scope of outdoor systems, GPS is probably the most widely known positioning system. Unfortunately, GPS signal is not available for indoors. Signals are blocked by most of the walls. The US Federal Communication Commission has mandated that all wireless phone providers develop a way to locate any phone in case of emergency. E911 is not a proper location sensing system but this initiative has started the interest of many companies to develop a variety of location systems to determine a cellular phone location (Federal Communications Commission, 2001). Location information is valuable in many areas, including security, RF coverage assessment, network management and information for users. Since accurate location information as a service of WLANs is a recent feature, some of the more sophisticated applications are just emerging. There are some existing commercial technologies that provide location sensing on a Wireless LAN that we have used in our experiments during the research process and we will mentioned below.

- **Ekahau** (Ekahau, 2011) is a Finnish company that provides software solutions for location sensing on a Wireless LAN. The Ekahau system has been specifically developed to provide GPS-like positioning inside buildings and to a certain extent outside buildings over wireless networks. The system works over different WLAN standards such as 802.11 and HiperLAN. Ekahau's positioning software suite (named Ekahau Positioning Engine 2.0) includes three sub-components, the Ekahau Client, Ekahau Positioning Engine and Ekahau Manager. To be 'trackable' each mobile client (Laptop or PDA) must be running the client software. The Ekahau Positioning Engine is a Java-based server software that is used for the calculation of the mobile client locations. The Ekahau Manager displays maps used for positioning, allowing visual tracking of devices, and analysing accuracy and performance of the software. Currently the supported Operating Systems are Windows 2000/XP and Pocket PC 200X.

  Before the Positioning Engine can be used for tracking devices over the WLAN, it needs to be calibrated manually. A floor map is first uploaded on which the Positioning Engine draws tracking rails to create a positioning model. The calibration of the system requires the user to physically move around the area with a mobile client equipped with the Ekahau Client. Approximately every ten feet the current position has to be indicated by clicking on the uploaded map on the client to record sample points containing received signal strength samples. The software does not need the location co-ordinates of the access points (Ekahau, 2011). Tracking of mobile clients on the WLAN can be initiated after calibrating the system though the Ekahau Manager. Real time location of users is shown on the provided floor plan. To locate clients, the software calculates the signal strengths from the access points and compares the m to calibrated signal strength samples and to the map of the location. Increasing the number of access points increases the accuracy of the system. Ekahau claims to have up to 1 meter (3.5 ft) average accuracy of locating a mobile client.

- **Newbury Networks** (Networks, 2011) is a Boston based company, providing solutions for Wireless LAN security. Their latest offering "WiFi Watchdog" is an advanced location-based software solution that secures wireless networks against intruders and rogue users. The system works with a standard WLAN setup to locate and monitor authorized and unauthorized traffic on the 802.11 network as well as get information on rogue access points.

  Newbury Networks has another location-based product called Digital Docent, which provides location based content on a Mobile Client. Such information can be useful in museums, exhibitions and education campuses by providing dynamic information based on a user's current location. Both of these applications are based on a core technology called LocaleServer. One of the key capabilities of LocaleServer is the ability to detect locations of wireless devices within range of the WLAN. The advantage that LocaleServer has above traditional networking tools is that it can detect all 802.11 traffic within range of the network, not just the traffic on the network. LocaleServer can also be used for tracking and monitoring all 802.11 devices and equipment (Networks, 2011). Since this system does not require a "calibration" step unlike Ekahau's Positioning Engine, the technology used for sensing locations is also different. Detailed technical information for the LocaleServer was unavailable. A probable solution this might use is triangulation, using wireless signal strengths and signal phase.

- **Aruba Networks** (Aruba Networks, 2007) is an American company which offers a mobility solution allowing the secure access to data, voice and media streaming services using a wireless infrastructure. It has a location tracking solution which uses an enterprise wide WLAN deployment to provide precise location tracking of any Wi-Fi device in the research facility. The RF Locate application can track and locate any Wi-Fi device within range of the Aruba mobility infrastructure. Using accurate deployment layouts and triangulation algorithms devices can be easily located include PDAs, rogue APs/Clients, VoWLAN phones, laptops, Wi-Fi asset management tags.

  Although many alternatives exist, most successful indoor location techniques are based on the RSSI triangulation method. But basic RSSI triangulation does not provide sufficient accuracy for many of the users of location information. While techniques such as analysis of building material and walk around calibration can improve the accuracy of RSSI measurements, they add considerable expense and complexity to the network installation, and the improvement in accuracy erodes over time, as the environment changes. WLANs are cellular, where neighbouring APs operate on different RF frequencies (channels) to avoid interference. The Wi-Fi medium access control layer (MAC) allows any station in a basic service set to transmit at any time. Therefore all stations (including the AP) should be listening on the cell's RF channel all the time, to avoid missing transmissions.

The above said, explained the use of time-stealing APs to monitor other channels while nominally providing coverage of their own cell. An alternative technique is to deploy dedicated RF monitors named "Air Monitors" (AMs). AMs are identical to APs (the same hardware and software), but they are configured permanently in 'listening' mode. This is a very useful capability, because the AMs contribute not just to location accuracy but also by improving security coverage, detecting RF sources that may be security risks or interferers. The drawback of using dedicated AMs is that they add to the capital costs of the network.

In addition to the raw contextual information such as location, noise level and temperature, we are also interested in high-level context information such as the user's "current activity". It is, however, a big challenge to sense complex social contexts. One approach is machine vision, based on camera technology and image processing. Another possible approach is to consult the user's calendar directly to find out what the user is supposed to do at certain time. The user, however, is not always willing or able to put her activities in the calendar and she may not always follow the calendar. A third method is to use Artificial Intelligence techniques to recognize complex context by combining several simple low-level sensors (Schmidt et al., 1999c). The TEA project illustrates how to recognize whether the phone or PDA is in hand, on a table or in a suitcase, by using three sensors: light, and acceleration in two directions (X and Y) (Schmidt et al., 1999b). The idea of the learning architecture deployed in TEA is to cluster the input signals with a neural network called the 'Kohonen Self-Organizing Map' (KSOM). The outcome of this clustering is then labelled or classified, after which it is checked by a probabilistic finite state machine (Van Laerhoven, 1999).

2.1.3 Context modelling

As indicated in (Henricksen et al., 2002), there is usually a significant gap between sensor output and the level of information that is useful to applications, and this gap may be bridged by various types of context information processing. For example, a location sensor may supply raw coordinates, whereas an application might be interested in the identity of the building or room a user is in. Applications usually represent the context information as a set of designer-defined descriptors (e.g. numerical values of time or device IDs). While this format might be appropriate for developers, it could be difficult for the users (who are used to handling higher-level semantic descriptors) to understand. When modelling context, it is necessary to distinguish among different types of context information, so that a taxonomy is useful to appropriately model the concepts. However, most existing approaches do not define such a taxonomy. Instead, they use a "generic" context, no matter what type of context information they are dealing with (Ayed et al., 2007; Henricksen et al., 2002; Sheng & Benatallah, 2005; Vildjiounaite & Kallio, 2007). Some taxonomies have been proposed such as (Ardissono et al., 2007; Goker & Myrhaug, 2002; Pires et al., 2005; Schmidt, 2005) and when they are compared, the following is found. Some types of context, such as the social context or the physical context, are common to nearly all the proposals, while others like the computational context only appear in some of them. This is because most of these classifications are focused on specific domains. For example, (Ardissono et al., 2007), make their classification by bearing in mind the management of context-aware web service-based workflow systems, and Schmidt (Schmidt, 2005) only considers contexts that are relative to e-learning systems in order to consider the learners situation in an appropriate manner. Moreover, the same information is considered to be of different context types, such us the role of a user or the bandwidth communication cost in (Goker & Myrhaug, 2002) and (Pires et al., 2005).

There are several classifications for context modelling. We follow the (Strang & Linnhoff-Popien, 2004) approach where they are classified by the scheme of data structures which are used to exchange contextual information in the respective system

2.1.3.1 KeyValue models

The model of key-value pairs is the most simple data structure for modelling contextual information. Already (Schilit et al., 1994) used key-value pairs to model the context by providing the value of a context information (e.g. location information) to an application as an environment variable. The key-value modelling approach is frequently used in distributed service frameworks, e.g. Capeus (Samulowitz et al., 2002). In such frameworks, the services itself are usually described with a list of simple attributes in a key-value manner, and the employed service discovery procedure, e.g. SLP, Jini, see (Strang & Linnhoff-Popien, 2003) operates an exact matching algorithm on these attributes. In particular, key-value pairs are easy to manage, but lack capabilities for sophisticated structuring for enabling efficient context retrieval algorithms.

2.1.3.2 Markup scheme models

Common to all markup scheme modelling approaches is a hierarchical data structure consisting of markup tags with attributes and content. In particular, the content of the markup tags is usually recursively defined by other markup tags. Typical representatives of this kind of context modelling approach are profiles. They usually base upon a serialization of a derivative of Standard Generic Markup Language (SGML), the superclass of all markup languages such as the popular XML. Some of them are defined as extension to the Composite Capabilities / Preferences Profile (CC/PP) (World Wide Web Consortium (W3C), 2007a) and User Agent Profile (UAProf) (WAP Forum, 2007) standards, which have the expressiveness reachable by RDF/S and a XML serialization.

These kinds of context modelling approaches usually extend and complete the basic CC/PP and UAProf vocabulary and procedures to try to cover the higher dynamics and complexity of contextual information compared to static profiles. An example of this approach are the Comprehensive Structured Context Profiles (CSCP) by (Held et al., 2002). Unlike CC/PP, CSCP does not define any fixed hierarchy. It rather supports the full flexibility of RDF/S to express natural structures of profile information as required for contextual information. Attribute names are interpreted context-sensitively according to their position in the profile structure. Hence, unambiguous attribute naming across the whole profile as necessary with CC/PP is not required. Another drawback of CC/PP, the restricted overriding mechanism of default values only, replaced by a more flexible overriding and merging mechanism, allowing for instance to override and/or merge a whole profile subtree.

A similar approach to CSCP is the CC/PP Context Extension by (Indulska et al., 2003). They extended the basic CC/PP and UAProf vocabulary by a number of component-attribute trees related to some aspects of context, e.g. concerning location, network characteristics, application requirements, session information as well as certain types of relations and dependencies. The authors concluded that the their approach is capable of enabling context-awareness to applications and other parts of ubiquitous computing infrastructure. They already realized, that it is difficult and non-intuitive to capture complex contextual relationships and constraints due to the underlying CC/PP.

Another context modelling approach in the markup scheme category – which does not bear towards CC/PP – is the Pervasive Profile Description Language (PPDL) (Chtcherbina & Franz, 2003). This XMLbased language allows to account for contextual information and dependencies when defining interaction patterns on a limited scale. The number of evaluable contextual aspects and the comprehensiveness of the language itself seems to be relatively limited. Due to the fact that no design criteria and only parts of the language are available to the public, the actual appropriateness of this context modelling approach remains unknown.

There are several other context modelling approaches in the markup scheme category. They are oftentimes either proprietary or limited to a small set of contextual aspects, or both. Examples affected by these limitations are, among others, the context configuration of Capra et al.'s reflective middleware (Capra et al., 2001) the Centaurus Capability Markup Language (CCML) (Kagal et al., 2001), ConteXtML (Ryan, 1999) or the note-tags of the stick-e notes system (Brown et al., 1997).

2.1.3.3 Graphical models

A very well known general purpose modelling instrument is the Unified Modelling Language (UML) which has a strong graphical component (UML diagrams). Due to its generic structure,UML is also appropriate to model the context. This is shown for instance by Bauer in (Bauer et al., 2003), where contextual aspects relevant to air traffic management are modelled as UML extensions. Another example is the nicely designed graphics oriented context model introduced in (Henricksen et al., 2003), which is a context extension to the Object-Role Modelling (ORM) approach (Halpin et al., 2008) according some contextual classification and description properties (Henricksen et al., 2002). In ORM, the basic modelling concept is the fact, and the modelling of a domain using ORM involves identifying appropriate fact types and the roles that entity types play in these. Henricksen extended ORM to allow fact types to be categorised, according to their persistence and source, either as static (facts that remain unchanged as long as the entities they describe persist) or as dynamic.

The latter ones are further distinguished depending on the source of the facts as either profiled, sensed or derived types. Another quality indicator introduced by Henricksen is a history fact type to cover a time-aspect of the context. The last extension to ORM made by Henricksen for context modelling purposes are fact dependencies, which represent a special type of relationship between facts, where a change in one fact leads automatically to a change in another fact: the depends on relation. This kind of approach is particularly applicable to derive an ER-model (Chen, 1976) from it, which is very useful as structuring instrument for a relational database in an information system based context management architecture such as the one described in (Indulska et al., 2003).

2.1.3.4 Object oriented models

Common to object oriented context modelling approaches is the intention to employ the main benefits of any object oriented approach —namely encapsulation and reusability— to cover parts of the problems arising from the dynamics of the context in ubiquitous environments. The details of context processing is encapsulated on an object level and hence hidden to other components. Access to contextual information is provided through specified interfaces only A representative for this kind of approach are the cues (Schmidt et al., 1999c) developed within the TEA project (Schmidt & Van Laerhoven, 2001). The concept of cues provides an abstraction from physical and logical sensors. A cue is regarded as a function taking the value of a single physical or logical sensor up to a certain time as input and providing a symbolic or sub-symbolic output. A finite or infinite set of possible values is defined for each cue. The output of each cue depends on a single sensor, but different cues may be based on the same sensors. The context is modelled as an abstraction level on top of the available cues.

Thus the cues are objects providing contextual information through their interfaces, hiding the details of determining the output values. Another approach within the object category is the Active Object Model of the GUIDE project (Cheverst et al., 1999). Again, the chosen approach has been primarily driven by the requirement of being able to manage a great variety of personal and environmental contextual information while maintaining scalability. All the details of data collection and fusing (e.g. the context adaptive composition of HTML fragments) are encapsulated within the active objects and thus hidden to other components of the system.

The approach of Bouzy and Cazenave (Bouzy & Cazenave, 1997) followed a similar intention: They propose to use general object oriented mechanisms to represent contextual knowledge about temporal, goal, spatial and global contexts in computer Go (a 4000 years old game that is very famous in Japan, China and Korea). They justified their object oriented context modelling approach with its inheritance and reutilization capabilities, allowing "to define the smallest number of properties, functions and rules [..] in order to simplify knowledge representation in very complex domains and systems".

2.1.3.5 Logic-based models

A logic defines the conditions on which a concluding expression or fact may be derived (a process known as reasoning or inferencing) from a set of other expressions or facts. To describe these conditions in a set of rules a formal system is applied. In a logic based context model, the context is consequently defined as facts, expressions and rules. Usually contextual information is added to, updated in and deleted from a logic based system in terms of facts or inferred from the rules in the system respectively. Common to all logic based models is a high degree of formality. One of the first logic based context modelling approaches has been researched and published as Formalizing Context in early 1993 by McCarthy and his group at Stanford (Mccarthy, 1993),(Mccarthy & Buva, 1997). McCarthy introduced contexts as abstract mathematical entities with properties useful in artificial intelligence.

He prevented emphatically to give a definition what context is. Instead he tried to give a formalization recipe which allows for simple axioms for common sense phenomena, e.g. axioms for static blocks worlds situations, to be lifted to context involving fewer assumptions, e.g. contexts in which situations change. Thus lifting rules, which relate the truth in one context to the truth in another context, are an important part of the model itself. McCarthy's model already supports the concept of inheritance (Mccarthy & Buva, 1997) (De Bruijin, 2003).

The main focus of Giunchiglia's approach, sometimes referred to as Multicontext Systems, is less on context modelling than on context reasoning (Ghidni & Giunchiglia, 2001), (Giunchiglia, 1993). He take a context to be that specific subset of the complete state of an individual entity that is used for reasoning about a given goal; it is seen as a (partial) theory of the world which encodes an individual's subjective perspective about it.

Another early representative of this kind of approach is the Extended Situation Theory by Akman and Surav (Akman & Surav, 1997). As the name implies it extends the Situation Theory which has been proposed by (Barwise & Perry, 1981). Barwise and Perry tried to cover model-theoretic semantics of natural language in a formal logic system. Akman and Surav used and extended this system to model the context with situation types which are ordinary situations and thus first-class objects of situation theory. The variety of different contexts is addressed in form of rules and presuppositions related to a particular point of view. They represent the facts related to a particular context with parameter-free expressions supported by the situation type which corresponds to the context.

A similar approach is the Sensed Context Model proposed by (Gray & Salber, 2001). They use first-order predicate logic as a formal representation of contextual propositions and relations. Another approach within this category is the multimedia system by (Bacon et al., 1997). In this system the location as one aspect of the context is expressed as facts in a rule based system. The system itself is implemented in Prolog.

2.1.3.6 Ontology-based models

Ontologies are a promising instrument to specify concepts and interrelations (Uschold & Gruninger, 1996), (Gruber, 1993). They are particularly suitable to project parts of the information describing and being used in our daily life onto a data structure usable by computers. One of the first approaches of modelling the context with ontologies has been proposed by (Öztürk & Aamodt, 1997). They analysed psychological studies on the difference between recall and recognition of several issues in combination with contextual information. From this examination they derived the necessity of normalizing and combining the knowledge from different domains. They proposed a context model based on ontologies due to their strengths in the field of normalization and formality. Another approach within the ontology category has been proposed as the Aspect-Scale-ContextInformation (ASC) model (Strang & Linnhoff-Popien, 2003).

Using ontologies provides an uniform way for specifying the model's core concepts as well as an arbitrary amount of subconcepts and facts, altogether enabling contextual knowledge sharing and reuse in an ubiquitous computing system (De Bruijin, 2003). This contextual knowledge is evaluated using ontology reasoners. The model has been implemented applying selected ontology languages. These implementations build up the core of a non monolithic Context Ontology Language (CoOL), which is supplemented by integration elements such as scheme extensions for Web Services and others (Strang et al., 2003a), (Strang et al., 2003b). Beyond determination of service interoperability in terms of contextual compatibility and substitutability, this language is used to support context-awareness in distributed service frameworks for various applications. For instance a contextual motivated non-carrier service handover is presented as one of the applications (Strang et al., 2003c).

The CONON context modelling approach (Wang et al., 2004) is based on the same idea of the ASC/CoOL approach, namely to develop a context model based on ontologies because of its knowledge sharing, logic inferencing and knowledge reuse capabilities. Wang et al. created an upper ontology which captures general features of basic contextual entities and a collection of domain specific ontologies and their features in each subdomain. The CANON ontologies are serialized in OWL-DL which has a semantic equivalence to well researched description logics. This allows for consistency checking and contextual reasoning using inference engines developed for description languages.

A promising emerging context modelling approach based on ontologies is the CoBrA system (Chen et al., 2003). This system provides a set of ontological concepts to characterize entities such as persons, places or several other kinds of objects within their contexts. The CoBrA system uses a broker-centric agent architecture to provide runtime support for context-aware systems, particularly in Intelligent Meeting Rooms, a prevalent scenario of an ubiquitous computing environment.

2.1.4 Context- aware system architectures

In order to support the development and to ease the implementation of context-aware systems, many architectures were proposed with characteristics related to the application itself. Devices in a AmI systems have a main characteristic and it is their context awareness which allows them to provide proactively adapted services to user and to applications according to the global context domain and techniques used. The architecture design is an important step in the development of context-aware systems. System architecture is created early in the development process and permits the creation of a high level design of the system which takes into account the fulfilment of requirements's implementation. Implementation depends on conditions like e.g. the amount of users, the available resources of the used devices (PCs or small and handheld devices) or the facility of the system.

According to (Baldauf et al., 2007) there are three different approaches based on the above considerations. The approach of Direct Sensor Access is often used in devices with sensors locally built although it is not suitable for distributed systems due to its natural direct access. The Middleware-based design uses methods of encapsulation and introduces a layered architecture to context-aware systems. The other logical approach is to permit multiple client access to remote data sources in a Context Server. This distributed approach (Baldauf et al., 2007) extends the middleware based architecture by introducing an access managing remote component. There exist many researches on the field of architectures, frameworks and middlewares for context-aware systems. We can highlight and compare the following systems:

- The Active Badge system (Want et al., 1992) is based on a distributed architecture of sensors. It has a layered architecture of an application running on the server and it is composed of the following four layers: the network controller which supervises the operation of the sensor network; the information presentation which is responsible for data management and control of localization information; the data processing which selects the interesting information at the time of localization variation; and the user interface to display the textual information about badges variation position. The Active Badge is specific to localization systems and cannot be easily used for other kind of context-aware systems. Finally it does not make any abstraction of location information which makes it very dependent to the hardware infrastructure.

- In the same line is the Xerox project ParcTab (Want et al., 1996) that has a material infrastructure that enhances the development of applications aware to localization context (person location, surrounding devices, nearby people, etc.). The ParcTab is a primitive localization-aware system based on hardware infrastructure like the Active Badge. The software architecture is very dependent on hardware and does not provide a good abstraction of contextual information.

- Another one is Stick-e-notes project (Pascoe, 1997) that is a framework to support the development of context-aware application where localization is the basic component of context. The architecture is divided into four components: SEPREPARE (enables the user to prepare notes); SEMANAGE (permits the management of notes); SETRIGGER (enables notes triggering whenever similar context appears) and SESHOW (enables the display of triggered notes and their storage). This project use a limited set of location information. It is hardware dependent and do not provide a significant improvement of context abstraction which is similar to ParcTab.

- Besides, we have Cyberguide (Abowd et al., 1997) which architecture is composed of the following elements: An electronic geographical card of the physical environment visited by the tourist; a browser that permits the detection of the tourist's current location; a messenger which provides a message delivery service to the tourist to send request, suggestion, communication with other tourists and to receive broadcasted messages. It is also specific to localization systems, it does not interpret contextual information in a higher level of abstraction, it is very dependent to the hardware used and does not offer an extensible and reusable software architecture.

- On the other hand, we have the CASS tool (Fahy & Clarke, 2004) that is a middleware for supporting the development of context-aware applications. It does provide a good abstraction of contextual information and use an object oriented model for context description. The architecture is based on a server containing a database of contextual information and a knowledge base with an inference engine to infer other contextual information using a back chaining mechanism. CASS also provides a good abstraction of context due to its interpretation module and a reasoning mechanism which makes it more proactive however the centralized architecture is its weakness because if the server is down all the system will be affected and becomes non operational.

- There is also CORTEX (Biegel & Cahill, 2004) that proposed a framework to ease the development of context-aware mobile applications. The architecture is based on the "sentient object" which has some beneficial characteristics for pervasive computing environment such as: sensitivity, autonomy and proactivity. The architecture is composed of three modules: a module for fusion and interpretation of contextual information; a module for a hierarchical representation of context; and an inference engine which specifies the applications behaviour to a given context and uses the execution model event-condition-action. The discovery mechanism is not well detailed by authors and does not allow a measuring of the extensibility of the architecture. Also, the model of context used does not provide a complete set of contextual information needed for adaptation task.

- The context toolkit (Dey & Abowd, 1999) was proposed as a tool to help the developers of context-aware systems. It has a layered architecture that permits the separation of context acquisition, representation and adaptation process. It is based on context widgets which operate similarly to graphical user interface widgets in order to hide the complexity of physical sensors. These widgets offer a good abstraction of context and provide reusable blocs for context sensing. This architecture offers a distributed communication among system devices and reusable widgets but the discovery mechanism is centralized which does not make it a perfect peer-topeer communication model. It has a limited extensibility when the number of devices increases. The architecture does not contain a layer or a module for context reasoning because the model used for context representation (key/value) does not permit a good reasoning.

- Another one is Hydrogen (Hofer et al., 2003) is an architecture and a framework for context-aware systems. It is a three layered architecture that responds to particular requirements of mobile devices. The architecture has the following layers: adaptation, management and application. The context server (management layer) contains all the sensed information perceived by the sensors of the adaptor layer and provides context to the application layer of the attached device or other devices using a peer-to-peer communication model. The Hydrogen approach considers context as any pertinent information on an application environment and describes it using an object oriented model. Also, it makes it very dependent to sensors. The architecture does not contain a reasoning module on context to ease the adaptation task.

- SOCAM (Gu & Pung, 2004) is an architecture of a service oriented context-aware middleware for building and rapid prototyping of context-aware mobile services in an intelligent car. The architecture is composed of the following components: context provider; context interpreter; context knowledge and context reasoner; service locating service; context-aware mobile service and context database. The architecture uses the client/server model where the context interpreter collects contextual information from context providers and context database and provides them to the context aware mobile services and the service locating service. The main strength of the SOCAM architecture is its context reasoner which uses ontology for context description and allows a robust reasoning on context. It uses two classes of ontologies: domain specific and generalized ontologies. Also, several reasoning systems can be incorporated in the context interpreter to support a variety of reasoning tasks. The architecture was proposed to support a single distributed application which limits its use. In addition to the major problem of a centralized architecture that contradicts the nature of a pervasive system which is a distributed one with autonomous devices.

- CoBrA (Chen, 2004) is an architecture based on broker agent to support the development of context-aware applications in an intelligent space. The broker is an autonomous agent that manages and controls the context model of a specific domain. It runs on a dedicated computer (server) with powerful resources. The broker agent has a layered architecture containing the following components: context knowledge, context reasoner engine, context acquisition module and privacy management module. CoBrA uses ontology for context description which allows a good reasoning and a better sharing of contextual information. It uses a centralized model for the storage and the processing of context in order to save the limited resources of mobile devices and uses a confidentiality policy for the user. The architecture requires a dedicated server for the broker which increases its cost and limits its usability in addition to the problem of a centralized architecture.

- Appear Networks (Appear Networks, 2008) is is another context-aware commercial platform designed to provide contextual information to employees and informative clients. Appear consists of the context engine and its modules, which are: provisioning module, device management module and the synchronization Module. The Context Engine implements a rules engine, which determines what should be available to whom, and where and when it should be available. The architecture of the system is modular and separates the system responsibilities into: server, one or more proxies, and client software installed in the wireless devices. The server is the central point of administration and coordination of proxies. It is where the contextual information is managed in a centralized and global way. In fact, it administers the applications distributed by the platform. The local proxies are installed in each interest point and they are in charge of handle the communication with clients and server.

The approach depends on special requirements and conditions such as the location of sensors (local or remote), the amount of possible users (one user or many), the available resources of the used devices (high-end-PCs or small mobile devices) or the facility of a further extension of the system.But to consider context part of the knowledge of the domain, it needs to be formally represented and managed in order to reason about it. Knowledge representation and reasoning issues will be described in next section. Furthermore, the method of context-data acquisition is very important when designing context-aware systems because it predefines the architectural style of the system at least to some extent. Chen (2004) presents three different approaches on how to acquire contextual information. Direct sensor Access; Middleware infrastructure; Context server; In this paper we will focus on middleware based and context-server based systems with regards to their usability in distributed systems. Many layered context-aware systems and frameworks have evolved during the last years. Most of them differ in functional range, location and naming of layers, the use of optional agents or other architectural concerns. Besides these adaptations and modifications, a common architecture in modern context-aware applications is identifiable when analysing the various design approaches.

Although these approaches successfully deal with the implicit input provided by the awareness, but they do not provide a general architecture that can be applied to different scenarios and they do not neither provide a self-adaptation process. We address this problem by designing a generic system using Appear Platform (Appear Networks, 2008) and also applying evaluation techniques from the user's point of view in order to provide with a user feedback into the system (Sánchez-Pi & Molina, 2009b) and (Sánchez-Pi & Molina, 2010b). Details are given in Section 4.1.

2.2 Knowledge representation and reasoning

The representation of knowledge in knowledge based systems is varied. For instance, the development of semantic web technology enables the information on the current web to have precise meaning and machine-interpretable form, that would allows computers and people processing the same data to have a common understanding of what the terms means (Berners-Lee & Miller, 2002). Semantic web is also used in KBS development through ontologies that enable the construction of KBSs through reusable components across domains and tasks (Gomez-Perez & Benjamins, 1999). Ontologies are used to represent domain knowledge in knowledge-based programs. This is achieved using formal declarative representations of the domain knowledge; that is sets of objects and their describable relationships (Gruber, 1995). Researchers in the area of knowledge modelling have started to realize the importance of ontology in developing domain models since the underlying principle of modelling is to achieve agreed representations in a unified manner for the domains in which they are investigating. Through the use of semantic web languages such as DAML+OIL, SHOE and RDF, ontologies can be described and these descriptions are used to create the knowledge base of the KBS (Noy et al., 2001). This allows the KBS developer to focus on domain knowledge representation instead of mark up tags and correct syntax to build KBS faster.

Because of the increasing development of these centralized systems and its applications, the development of knowledge-based systems (KBS) are also being increased as the way of modelling and manage knowledge in context-aware systems. KBS are advanced systems of complex problems representation. Its architecture and representation formalisms are the base of nowadays systems. The nature of the knowledge is usually derived from the experience in specific areas and its validation requires a different methodology of the one used in the conventional systems because the symbolic characteristic of the knowledge.

2.2.1 Methodologies

There are several methodologies for the development of KBS but the most known is KADS (Knowledge Acquisition and Document Structure) and its successor CommonKADS (Schreiber et al., 1999), which is becoming the de facto European standard. CommonKADS offers six models to represent context: organization, tasks, agents, communication, knowledge and design. The knowledge model is a very good approach to the knowledge representation (Wielinga et al., 1994). It describes in three categories (domain knowledge, tasks knowledge and inferences knowledge) what a specific agent has and what is relevant for the development of a particular task describing the structure based on its use.

MIKE (Model–based and Incremental Knowledge Engineering) (Angele et al., 1996) is one of them that provides a methodology for the development of KBS following all the aspects of the process, from the knowledge acquisition. It provides a structured life-cycle facilitating the maintenance, the depuration and the reuse of the resulting product. Protegé (Eriksson et al., 1995; Puerta, 1991; Tu et al., 1995) is another one that generates knowledge acquisition tools using reusable libraries. VITAL (Domingue et al., 1993; Shadbolt et al., 1993) is a less known methodology but it has similar points to CommonKADS and MIKE. It for instance, produces four products by result of its application. The Conceptual Model it's similar to CommonKADS and the Design Model it's similar to MIKE. We have chosen CommonKADS to model our problem because it covers the main points of the development of a KBS, from the very start analysis (in order to identify the problem and to establish the suitability of the solution based on a KBS) to the implementation of the project.

CommonKADS is a structured approach to the development of knowledge based systems and as such is to be seen in contrast to unstructured approaches such as rapid prototyping. It does not require a commitment to any specific implementation paradigm. According to CommonKADS, the development of a knowledge-based system is to be seen as a modelling process, during which models of acquired knowledge at different levels of abstraction are developed.

There are three levels of models:

- The process level identifies the tasks involved in the domain, the nature of data flows and stores, and the assignment of ownership of tasks and data stores to agents.
- The system level, the co-operation model describes in detail the interactions between the system and external agents, and how the internal agents interact. The co-operation model is used to separate the user task model from the system task model, and allows the knowledge wholly internal to the system to be readily identified.
- The expertise level corresponds to an expertise model. This divides the task of describing domain and expert knowledge and its use within the system into a number of supportive tasks. The layer-based framework for expertise consists of:
  - The domain layer is comprised of static or slowly changing knowledge describing concepts, relations, and structures in the domain.
  - The inference layer reformulates the domain layer in terms of the different types of inferences and reasoning that can be made.
  - The task layer defines knowledge about how to apply the knowledge in these two layers to problem-solving activities in the domain.
  - The strategy layer concerns selecting, sequencing, planning and repairing tasks

2.3 Multi-agent systems

Intelligent agents are computer systems capable of independent actions on behalf of its user (Wooldridge & Jennings, 1995). For a large class of the envisaged AmI applications, the added value of new services is likely to be for people in ordinary social contexts. Such applications beg for technologies that are transparent, so that their functional behaviour can be easily understood by users. Intelligent agents seem to be the appropriate solution for

AmI environments since they provide autonomy and proactivity. If we assume that agents are abstractions for the interaction within an AmI environment, one aspect that we need to ensure is that their behaviour is coordinated. For this purpose, we need decision rules that take into consideration the context (location, user profile, type of device, etc.) in which these interactions take place. Agents are sociable and that is the appropriate technology for communication and coordination. Taking care this, the system needs an organization similar to the one envisaged by intelligent agent societies. The society is there not only to regulate behaviour but also to distribute responsibility amongst the member agents. In (O'Hare et al., 2004), O'Hare et al. advocate the use of agents (as well as we do) as a key enabler in the delivery of ambient intelligence.

Entities (human or companies) are naturally included as actors in any information systems environment. While computer systems process data, information and knowledge are truly processed by humans. However, the amount of available online information is growing at such rate that people are not being able to cope with it. Even more, complex knowledge processing and communication are required to offer, provide and consume services on behalf of humans. We need actors for helping humans in their knowledge-related tasks. Actors that can some how understand people's emotions and rational behaviour, or that can at least attempt to process complex information on our behalf. This is the underlying foundation of the concept of agent, computer systems capable of independent actions on behalf of its user (Durfee & Rosenschein, 1994)

These agents are usually defined by mean of features that characterized its behaviour as (Maes, 1990a). An agent is a computational system living in a complex and dynamic environment. The agent is capable of not only sensing but also acting on behalf of the environment. It also intend to achieve a set of objectives through out these actions. One of the most accepted definitions (Wooldridge & Jennings, 1995), propose a set of essential attributes: autonomy, sociability, reactivity and proactivity. The usefulness of the agents implies that they has to be sensitive to the changes in the environment. This means that they need to communicate and exchange information with the surrounded world. Communication plays a key role in agent technology to provide interoperability, it has been addressed by the Foundation for Intelligent Physical Agents (FIPA) (IEEE FIPA Standards Committee, 1997), an international non-profit association of companies and organizations that produces specifications of generic agent technologies. FIPA is concerned only with how communication is carried out between agents, it provides an Agent Communication Language (ACL) and a Message Transport Service. This language (ACL) is divided in three levels: message level, content level and communication level that are detailed above:

- The type of interaction for the communication is decided in the message level. The main function is to supply an intention or interpretation that the sender link to the content of the message. These intentions that humans transmit in a dialogue has been studied for the theory of speech acts (Searle, 1969) and enumerated by FIPA in a set of communicative acts(FIPA Communicative Act Library Specification, FIPA00037):
  - Information: query_ if, query_ ref, subscribe, inform, inform_ if, inform_ ref, confirm, disconfirm and not_ understood.
  - Task distribution: request, request_ whenever, cancel, agree, refuse, failure.
  - Negotiation: cfp, propose, accept_ proposal, reject_ proposal.

- In the communication level is where low level parameter are defined like: the sender, the receiver, the transmission protocol, etc. These parameters constitute the message structure (FIPA ACL Message Structure Specification, FIPA00061).
- In content level it is where the misunderstandings of a content message are managed

But these misunderstandings are not exclusively solved following a single representation model. An objective description of the concepts and relationships of the domain of knowledge of the message is also required. This explicit and formal specification of a shared conceptualization is what we usually called ontology (Gruber, 1991). An ontology allows that a content of a message can be interpreted unambiguously and independently from the context.

In its most prevalent use in AI, an ontology refers to an engineering artefact, constituted by a specific set of terms vocabulary used to describe a certain world model. These terms often appear as predicates, actions and concepts (hierarchely ordered). (FIPA Ontology Service Specification FIPA00086)

Although exists several FIPA compliant agent platforms (JADEX, JASON, 2APL, Agent-Factory, Diet Agents, MADKIT, JACK), JADE (Java Agent Development Framework) (TILAB, 2010) is the most convenient option as it simplifies the implementation of multiagent systems through a a set of graphical tools that supports the debugging and deployment phases. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. The synergy between the JADE platform and the LEAP libraries allows to obtain a FIPA-compliant agent platform with reduced footprint and compatibility with mobile Java environments down to J2ME-CLDC MIDP 1.0. The LEAP libraries have been developed with the collaboration of the LEAP project and can be downloaded as an add-on of JADE from this same Web site. Furthermore JADE provides the three mandatory roles that have to be present into a FIPA compliant agent platform: AMS (Agent Management System) that controls access and use of the platform, DF (Directory Facilitator) that provides yellow pages service and ACC (Agent Communication Channel) that provides the Message Transport Service.

2.3.1 Agents definition

Software agents provide then an ideal mechanism for implementing heterogeneous and complex distributed systems. Such systems cannot be easily developed using traditional software technologies because of their limits in coping with distribution and interoperability. The agent technology is well suited for applications that are based on communication of loosely-coupled systems, this is the case of AmI applications.

A software agent is a computational process that operates in an environment and based their decisions on what to reason about the information it obtains from the environment. One of the most important decisions when designing multi-agent systems is to decide the information exchanged between agents. Some authors propose the exchange of raw data, while others propose the exchange of beliefs (Beliefs) of agents. Regardless of the type of information that agents need to exchange, this exchange should be done in a coordinated fashion. Coordination is defined as the process of managing interdependencies between different actors. Therefore, the coordination of different actors of the system is an essential task (Lesser, 1999). Coordination can be done implicitly (without communication) or explicitly (using media). Implicit coordination of agents is based on using rules in conformity with policies or rules to be followed by agents. The advantage of implicit coordination is that it requires communication and is faster, however, requires careful planning and preliminary design complete. Coordination becomes a partnership when it comes to actors who have a common goal in a negotiation when it comes to agents with individual goals, or competitive.

The agent paradigm defines how to build concurrent and distributed software systems, which have low coupling. To achieve this, it is assumed that an agent is an autonomous entity that acts according to its objectives and communicates with other agents via asynchronous messages. This development paradigm targeting agents, represents a new conceptual level of abstraction, you can also use the practices of object-oriented paradigm. A software agent is defined as a computational process that is capable of carrying out flexible autonomous decisions to achieve their goals (Wooldridge & Jennings, 1995). For flexible decisions, decisions means having the following characteristics:

- Reactivity: To allow agents to perceive and respond to changing environments to meet their goals, which means that officers should respond quickly to changes in the environment.
- Social ability: By which the agents interact with other agents. This social skill is accomplished through the use of shared whiteboards or message passing between agents.
- Proactive: On which the agents behave in a manner directed by goals and take initiative to achieve their design goals. Proactivity refers to the ability to anticipate future plans of the agents.

An agent is a computational process that has control over its internal state and behaviour. Gets the environment through sensors and acts according to its objectives as set at the stage of design. A multi-agent system can be defined as a network of software agents that work together and share social skills. This social skills, means that the agents not only perform intra-agent communication (internal agents), but also inter-agent (including agents). Multi-agent systems belong to the field of artificial intelligence, which aims to provide principles for building complex systems involving multiple agents and also the necessary mechanisms to coordinate individual behaviour of agents (Stone & Veloso, 2000). A multi-agent system, can also be defined as a set of software agents that work together and have the following characteristics:

- Lack of global vision: Each agent has access to only one piece of information that exists in the global system.
- Decentralized control: The control of the information is decentralized and each agent is responsible for its own control.
- Autonomy: agents are considered independent processes, so that they now choose to make decisions based on data obtained from the environment.

- Asynchronous subsystems: Each of the agents behave asynchronously and with low coupling between each other, not necessary to use or know the operating system, programming language and hardware platform for each of them.
- Need for cooperation: There is an expressed need for cooperation between system agents.

In the field of multi-agent systems, the sub may come from the spatial, temporal or functional information, resources or knowledge. Therefore, one of the main characteristics of the agents is the cooperation between them for better performance or to use additional capabilities that do not possess. For proper cooperation between agents is necessary to define what should be communicated, when to perform the communication and whom to contact. From the point of view of cooperation, there may be three different types of interactions: (1) sets targets interactions between agents, (2) complementary objectives interactions between agents and (3) contact interactions objectives.

Many researchers have claimed the use of multi-agent systems to develop systems to obtain data in a distributed manner, where control and / or resources may also be distributed. In these systems, the agents provide the benefits of scalability and modularity to perform distributed reasoning of the data.

2.3.2 Multi-agent systems architectures

Multi-agent architecture, determine the mechanisms used by an agent to react to stimuli, act and communicate (Mas, 2005). You can make the following classification architectures that are based on multi-agent systems:

1. Reactive agent architectures, are characterized by not having as central to a symbolic model reasoning. These systems respond to changes in the immediate environment of a from a series of rules. The aim of these systems is to make the argument without explicitly modelling knowledge. A reagent therefore has no symbolic world model and does not use complex logical reasoning. However, attempts to carry out a series of objectives in a complex and dynamic environment. In such architectures, intelligent behaviour emerges from simple rules of interaction.
2. Deliberative architectures: These are models that use symbolic knowledge representation, starting from an initial state and are able to generate plans for achieving their goals (Maes, 1990b). Deliberative agents explicitly represent the model of the world in a symbolic manner and decisions are carried out through such logical reasoning.These types of agents must solve two problems.First, what is known as transduction problem, which consists of transferring real-world information to a timely and accurate symbolic description to be used at the time.Second, the problem of representation and reasoning, which is symbolically represented, information about complex real-world entities and processes and how to make the agents reason with this information in an efficient way to obtain useful results.The main drawbacks of purely deliberative architectures (and in general any logical model) is that increasing the knowledge the problems become very difficult to resolve according to the logical schema.Some works combine these

architectures with other arguments such as (Corchado & Laza, 2003), who have proposed building deliberative agents using the technology of case-based reasoning and its application in e-commerce environment.

3. Hybrid architectures: They combine elements of reactive and deliberative architectures. N is based on models of practical reasoning and the need to seek quick reactions so as to reason about the following objectives.The first model-based or a hybrid architecture was the PRS system (Georgeff & Lansky, 1987; Ingrand et al., 1992) and emerged continuation IRMA, dMARS (D'Inverno et al., 2004), JACK (Busetta et al., 1999), Agentis (Kinny & Phillip, 2004), etc.

In addition to the existing classifications by type of multi-agent architecture (Deliberative, reactive or hybrid), you can perform a partition of the multi-agent systems basis of these four dimensions: homogeneous agents non-communicative; heterogeneous non-communicative; homogeneous agents communicative; heterogeneous communicative.

The difference between the homogeneous and heterogeneous agents, is based on the homogeneous agents all have the same internal structure, including its objectives, domain knowledge and the possible actions and they have the same procedure for selecting the actions. The only difference between them, are the entries in the environment information (sensors) and actions carried out based on those entries. In contrast, heterogeneous agents have different objectives and generally opposing each other, which usually translates into a behaviour based on competition rather than collaboration. On the other hand, both the homogeneous and heterogeneous agents may not have communication skills, understanding communication skills sending and receiving messages.

2.3.3 Model of Belief-Desire-Intention (BDI)

The agent architectures based on the model Belief-Desire-Intention (BDI) originated with Rational Agency project at Stanford University in late 1980 and this model corresponds to a deliberative architecture hybrid.It has its origins in the conceptual model described by (Bratman et al., 1998), which lies in the philosophical model of human reasoning, originally developed by (Bratman, 1999), which in turn reduces the explanation of complex human behaviour motivational stance (Dennett, 1989).The position of reasoning implies that the causes for actions are always related to human desires, ignoring other facets of human motivation for action potential. It also proposes use of a consistent, psychological concepts correspond fairly with the terms that people use to explain the behaviour. In a BDI model of agents, each agent has a set of beliefs (Beliefs), Wish (Desires) and intentions (Intentions), which internally represent the mental state of each of the agents. Beliefs model the states of the world, desires refer to choices between the possible states and intentions are commitments that are purchased to achieve specific states. Beliefs define the partial knowledge that an agent has about the world as observed facts. Desires represent states that the agent would like to obtain or achieve. Finally, intentions represent desires that the agent is committed to achieve. In an instant be consistent with each other. The model of practical reasoning, which are based on BDI architectures, it involves two major processes: (1) decide what goals to achieve and (2) to achieve these objectives. The first process is called deliberative and the latter as reasoning or final action.The deliberation process involves selecting targets, considering the different desires and intentions generate. The final argument involves carrying out appropriate actions, given the available resources and generate plans and turn them into actions. The intentions of the BDI agents have the following properties:

1. Once an intention has been taken by the agent tries to carry it out.
2. Once an intention has been adopted, the agent continues to be carried out or finds it impossible to complete.
3. The current intentions or intentions may preclude other options.
4. An agent can only adopt an intention if there are preconditions to achieve it.

To make the theory computationally tractable BDI agents, Rao and George simplified the theory and only explicitly represented beliefs, while the desires were limited to events that trigger predefined templates of plans and intentions were represented explicitly in the execution stack plans are executed. In the theory model of BDI agents, (Rao & Georgeff, 1995a), define the beliefs (Beliefs), Wish (Desires) and intentions (Intentions) and attitudes that represent possible states of the world. Therefore, a BDI agent has its own beliefs (about self and environment), desires (are the states that want to get) and intentions (which consist of the plans adopted.) In addition, each BDI agent maintains a repository of available plans, known as a library of plans that are executed on the basis of intentions selected. BDI agents respond to changes in their desires and beliefs that result from the perception of the environment and are packaged into data structures, representing either new beliefs or new desires. Agents respond to these changes by selecting the library plans plans for each of the changes and instantiating these plans as intentions. Understand the intentions of both plans, which in turn include actions and targets to be carried out with the possibility of adopting new plans. (Jo et al., 2004) provided a detailed explanation of the three terms used in the BDI agents model:

- Beliefs: The beliefs of an agent corresponding to the information that is acquired from the environment. Represent the knowledge of the world or the state of the world as the value of a variable, a relational database or as symbolic expressions of predicate logic. This knowledge may be incomplete or incorrect.
- Desires: Desires represent states that the agent would like to achieve. The desires or goals are an essential component of the system state, represent the desired end state. A desire or goal can be a variable value, the structure of a record or a symbolic expression in some logic. The common situation is that agents are unable to meet all your needs, therefore, should focus on a subset of their desires and allocate resources to achieve them. The desires or goals that the agent chosen trying to achieve, are known as intentions.
- Intentions: Intentions represent desires that the agent tries to perform. Intentions are a set of plans or procedures and may also be a set of threads of a process that seeks the goals of the system. The intentions require the use of the internal reasoning of the agent to select the action to perform. The desires and intentions are carried out through the execution of a series of actions and the series of actions carried out a plan defined.

Beliefs that can store an agent may be of the following types:

- Dynamics: These include information that varies in time.
- Static: These are beliefs that do not change their value during the life cycle of execution of the agent.
- Own: correspond to information obtained by the input of the agent.
- Of other agents: These are beliefs we have about the beliefs of other agents, for example: (Bank) the balance of the ATM-1 is 3000 euros.

On the other hand, (Georgeff et al., 1999), made the following internal division of work based on the BDI agents model:

1. Those who use general models of practical reasoning based on concepts BDI.
2. Work-based computational models in architecture IRMA (Intelligent Resource-Bounded Machine).
3. Which are based on the computational model used in PRS systems (Georgeff & Lansky, 1987; Ingrand et al., 1992), widely used in practice implemented solutions such as JACK (Busetta et al., 1999), JAM (Huber, 1999), Jadex (Pokahr et al., 2003) and Agentis (Kinny & Phillip, 2004).

One of the language-oriented programming BDI agents best known AgentSpeak, originally proposed by (Rao, 1996). AgentSpeak is based on logic programming languages and it is inspired by the PRS system (Georgeff & Lansky, 1987) and (Ingrand et al., 1992), the system dMARS (D'Inverno et al., 2004) and BDI logic. It aims to be a compromise between BDI theory and practical systems of BDI agents as PRS (Georgeff & Lansky, 1987), (Ingrand et al., 1992). As a language based on BDI model, language constructs are based on Beliefs, Goals and Plans. The architecture of an agent in AgentSpeak has 4 components: a belief base, a library of plans, a set of events and a set of intentions. In AgentSpeak agents react to events planning and executing events occur as a result of changes in the agent's beliefs or the goals. The best–known AgentSpeak interpreter is Jason (Hübner & Bordini, 2010), which is implemented using the Java programming language under the GPL.

DMARS system (D'Inverno et al., 2004) is a successor to the PRS-based BDI agent model. Is the agent-based system has been used more from a practical standpoint. The system of formal specification by Mark d'Inverno and represents the operational model of the system through plans. As Jadex also an event-based system. Literal beliefs are first-order logic, so a belief is expressed as a formula where the number of variables is zero. DMARS system distinguishes only between two types of objectives: Achieve and query. Intentions are ordered sequences of plans that have been activated and are created in response to an external event. Finally, an agent dMARS consists of the following components:

- A library of plans.
- A selection function of intentions.

- A selection function of the plans.
- A function of event selection.
- A selection function replacements.
- A function to determine the next branch of a plan.

Although the BDI model has been extensively studied, issues such as how much to commit an agent with intentions, remain an area of research today. The dilemma is essentially the problem of balancing pro-active behaviour (directed by objectives) and reactive (event driven). There must be a balance between the level of commitment to an agent and reconsideration of intentions (Kinny et al., 1996). It is obvious that an agent must sometimes rule out intentions for the following reasons: the agent concludes that the intention is not to be achieved, the intention has already been achieved or the reason (the precondition) to achieve the intent is no longer present. In this context, (Wooldridge, 2000) proposed three levels of commitments for staff: Commitment blind, single-minded and open-minded Commitment Commitment. Para (Wooldridge, 2000), the last level of commitment depends deliberative process undertaken and therefore it is important for an agent to reconsider its intentions from time to time, however, this involves a computational cost. Autonomous agents cooperate in order to improve their information and this cooperation is conducted through the exchange of messages. In the BDI-type multi-agent systems, cooperation is carried out by means of taking beliefs (Beliefs) and intentions (intentions) of some other agents. For example, in sensor networks visual perception of the agent in a given time (their beliefs) should be shared, using the message exchange with other agents, thus obtaining a distributed reasoning.

2.3.4 Agents for context-aware systems: Why agents?

Agents provide an excellent modelling abstraction for autonomous intelligent entities that exhibit human-like characteristics, including reasoning, proactivity, communication and adaptive behaviour. Moreover, AmI and context-aware systems beg for technologies that are transparent, so that their functional behaviour can be easily understood by users. Intelligent agents seem to be the appropriate solution for these environments since they provide autonomy and proactivity. If we assume that agents are abstractions for the interaction within an AmI environment, one aspect that we need to ensure is that their behaviour is coordinated. For this purpose, we need decision rules that take into consideration the context (location, user profile, type of device, etc.) in which these interactions take place. Agents are sociable and that is the appropriate technology for communication and coordination. Taking care this, the system needs an organization similar to the one envisaged by intelligent agent societies.

During the past two decades, with the increase in complexity of projects associate with software engineering, agent concepts that originated from Artificial Intelligence (AI) have been considered to devise a new paradigm for handling complex systems (Genesereth & Ketchpel, 1994; Shoham, 1990, 1993; Wooldridge, 1997; Wooldridge & Jennings, 1996; Wooldridge et al., 2000).

Agent-oriented software engineering (AOSE) paradigm represents an interesting means of analysing, designing and building complex software systems and it is quite suitable to the new software development requirements

There are several applications domains for agent technology. Current research has focused on agent and multi-agent implementations which support the ubiquitous provisioning of context-aware services to mobile devices. CONSORT (Kurumatani et al., 2004) is a multi-agent architecture including a middle agent that translate sensor based raw representation of the locations into a conceptual one using a location ontology in two application domains that model the geographical space in a cognitive way: an intelligent assistant at a museum and a wireless-LAN based location system. Another is SMAUG (Nieto-Carvajal et al., 2004) which is a multi-agent context-aware system that allows tutors and pupils of a university to fully manage their activities. SMAUG offers its users with context-aware information from their environment and also gives them a location service to physically locate every user of the system. Another one is IUMELA (McGovern et al., 2007), is an intelligent modular-education learning assistant designed using a multi-agent systems in order to assist students in their career decision-making process. Another clear application was presented for supporting virtual elderly assistance communities, in the context of the TeleCARE project (Afsarmanesh et al., 2004). SMAUG (Nieto-Carvajal et al., 2004) is a multi-agent context-aware system that allows tutors and pupils of a university to fully manage their activities managing context-aware information from their environment. AmbieAgents (Lech & Wienhofen, 2005) proposes an agent-based infrastructure for context-based information delivery for mobile users. There is also a case study consisting in solving the automation of the internal mail management of a department in a university using the ARTIS agent architecture (Bajo et al., 2008).

But it is rather evident that only few efforts attempt to address the engineering issues related to agent technology as a whole. Most publications explore the future potentials of agents or describe implementations, leaving many fundamental engineering issues unaddressed. Little effort has been spent towards both evaluation methodologies and software tools for integrating and automating the evaluation process.

Even in the few foundational works that describe interesting methodologies for designing and developing agent-based systems (e.g. (Bresciani et al., 2004; DeLoach, 1999; Jonker et al., 2004; Wooldridge et al., 2000), no reference is made on one of the most important aspects of agent software engineering, namely formal performance evaluation.

All the above research projects although successful to their own problem domain, do not provide a generic architecture that can be applied to different scenarios and case studies. We tackled this problem by designing a generic MAS architecture for AmI scenarios (Sánchez-Pi et al., 2007; Sánchez-Pi et al., 2010; Sánchez-Pi & Molina, 2010b).

Due to highly dynamic properties of the above introduced environments, the software system running on them have to face problems such as: user mobility, service failure, resources or goal changes which may happen in any moment. To cope with these problems, such systems must be autonomous and self-managed. These paradigms do not require software developers to be aware of all scenarios, it only involve users. That is why there is a need of special kind of system that will combine ubiquity, context-awareness, intelligence and natural interaction in an AmI environment. The system will also adapt not only to changes in the environment but also to the user requirements and needs. User will take the relevant role providing an evaluation of the system behaviour while using it or once it has been used.

As is the case, with any new software engineering paradigm, the widespread development of complex software systems based on Multi-Agent Systems (MAS)in the scope of AmI requires not only new models but also the identification of the software abstractions. MAS paradigm introduces a number of new abstractions when compared to more traditional systems. They may be used by software developers to more naturally understand, model and develop an important class of complex distributed systems. Accordingly new analysis and design methodologies are needed to effectively engineer such systems.

Several methodologies for the analysis and design of MAS have been proposed so far but few of them explicitly focus on organizational abstractions. Some agent-oriented methodologies are MASCommonKADS (Schreiber et al., 1999), Tropos (Bresciani et al., 2004), Zeus (Nwana et al., 1999), MaSE (DeLoach, 2004) or INGENIAS (Pavon et al., 2005). For the process of analysing and designing agent-oriented systems, we have selected Gaia methodology (Wooldridge et al., 2000). One of the main motivations to use Gaia for developing multi-agent systems is the necessity of viewing the system as a human organization, represented by a social community of agents, since the organization is not only a collection of roles, and Gaia provides this aspect in an intuitive way. Gaia helps in this task by defining organizational abstractions (environment, roles, interactions between roles, organizational structure and organizational rules). One problem with Gaia could be that it is a very high-level methodology. Gaia specifies how a society of agents collaborates to reach the goals of the system, and what is needed of each one to achieve them. This high level view is precisely what it is needed in this research. MaSE, MASCommonKADS and INGENIAS could be used to analyse our multi-agent system problem, but their own characteristics make the process more complex, since our main intention is to focus in the organizational abstractions mention before, without any object-oriented technique or specification language.

MAS-CommonKADS is a methodology for knowledge-based system that defines different models (agent model, task model, organizational model etc.) in the system life-cycle using oriented-object techniques and protocol engineering techniques. Tropos is a requirement-based methodology; Zeus provides an agent platform which facilitates the rapid development of collaborative agent applications; MaSE is an object-oriented methodology that considers agents as objects with capabilities of coordination and conversation with automatic generation of code and UML notation; and INGENIAS proposes a language for multi-agent system specification and its integration in the life-cycle, as well as it provides a collection of tools for modelling, verifying and generate agents' code.

2.4 Dialogue systems for context-awareness

AmI systems usually consist of a set of interconnected computing and sensing devices which surround the user pervasively in his environment and are invisible to him, providing a service that is dynamically adapted to the interaction context, so that users can naturally interact with the system and thus perceive it as intelligent. To ensure such a natural and intelligent interaction, it is necessary to provide an effective, easy, save and transparent interaction between the user and the system. With this objective, as an attempt to enhance and ease human-to-computer interaction, in the last years there has been an increasing interest in simulating human-to-human communication, employing the so-called multimodal dialogue systems (Wahlster, 2006).

A dialogue system (DS) can be understood as an automatic system able of emulating a human being in a dialogue with a person, with the aim that the system meets a certain functionality (usually providing some information or performing a certain task). By means of intelligent spoken interfaces it is possible to bring access to applications in which traditional input interfaces cannot be used (e.g. in-car applications, access to information systems for disabled people, etc).

The scheme used for the development of these systems of includes several generic modules that deal with multiple knowledge sources and that must cooperate to satisfy users requirements. Using this premise, a DS can be described in terms of the following modules. The Automatic Speech Recognition module (ASR) transforms the user utterance into the most probable sequence of words. The Language Understanding module provides a semantic representation of the meaning of the sequence of words generated by the ASR module. The dialogue Manager determines the next action to be taken by the system following a dialogue strategy. The Query Manager receives requests to the system servers, processes the information and returns the result to the dialogue manager. The Response Generator module receives a formal representation of the system action and generates a user response that can include multimodal information (video, data tables, images, gestures...). A Text to Speech Synthesizer (TTS) generates the audio signal transmitted to the user.

The practical implementation that we propose for the dialogue system architecture is based on two main components: an IVR (interactive voice response) server and a set of web servers. The IVR provides users with multimodal web pages following the VoiceXML standard3, the ASR and TTS interfaces, and VoIP and telephony technologies. Web servers connected to the IVR via Internet provide dialogue management facilities, grammars and system prompts, and the access to the different services provided by the system.

The task of the dialogue Manager (DM) in a spoken dialogue system is to control the flow of the dialogue and to decide which actions to take in response to user input. In order to complete these tasks, the DM needs a dialogue strategy that defines when to take the initiative in a dialogue, when to confirm receipt of a piece of information, how to identify and recover from recognition and understanding errors, and so forth. In order to complete the tasks described above and to decide what to say, the dialogue manager needs to track the dialogue history and update some representation of the current state of the dialogue. In addition, the DM needs a dialogue strategy that defines the conversational behaviour of the system, for example when to take the initiative in a dialogue or when to establish common ground. The performance of a spoken dialogue system is highly dependent on the quality of its DM strategy. Unfortunately, the design of a good strategy is far from trivial since there is no clear definition as to what constitutes a good strategy. User populations are often diverse, thus making it difficult to foresee which form of system behaviour will lead to quick and successful completion of the dialogue. In addition, the omnipresence of recognition errors leads to constant uncertainty about the true intention of the user. As applications reach greater levels of complexity, the process of designing robust dialogue strategies becomes very time-consuming and expensive.

Learning statistical approaches to model the different modules that compose a dialogue system has been of growing interest during the last decade (Young, 2002). Machine-learning approaches to dialogue management attempt to learn optimal strategies from corpora of real human-computer dialogue data using automated trial-and-error methods instead of relying on empirical design principles. Models of this kind have been widely used for speech recognition and also for language understanding (Esteve et al., 2003; He & Young, 2003; Levin &

Pieraccini, 1995; Minker et al., 1999; Segarra et al., 2001). Even though in the literature there are models for dialogue managers that are manually designed, over the last few years, approaches using statistical models to represent the behaviour of the dialogue manager have also been developed (Lemon et al., 2006; Levin et al., 2000; Torres et al., 2003; Williams & Young, 2007).

These approaches are usually based on modelling the different processes probabilistically and learning the parameters of the different statistical models from a dialogue corpus. Continuous advances in the field of spoken dialogue systems make the processes of design, implementation and evaluation of dialogue management strategies more and more complex. The motivations for automating dialogue learning are focused on the time-consuming process that hand-crafted design involves and the ever-increasing difficulty of the dialogue complexity. Statistical models can be trained from real dialogues, modelling the variability in user behaviours. Although the construction and parametrization of the model depend on the expert knowledge of the task, the final objective is to develop dialogue systems that have a more robust behaviour, better portability, and are easier to adapt to different user profiles or tasks.

The most extended methodology for machine-learning of dialogue strategies consists of modelling the human-computer interaction as an optimization problem using Markov Decision Process (MDP) and reinforcement methods (Levin et al., 2000; Singh et al., 1999). The main drawback of this approach is due to the large state space of practical spoken dialogue systems, whose representation is intractable if represented directly (Young et al., 2007). Partially Observable MDPs (POMDPs) outperform MDP-based dialogue strategies since they provide an explicit representation of uncertainty (Roy et al., 2000). However, they are limited to small-scale problems, since the state space would be huge and exact POMDP optimization is again intractable (Young et al., 2007). An approach that scales the POMDP framework for implementing practical spoken dialogue systems by the definition of two state spaces is presented in (Young et al., 2005). Other interesting approaches for statistical dialogue management are based on modelling the system by means of Hidden Markov Models (HMMs) (Cuayahuitl et al., 2005) or using Bayesian networks (Meng et al., 2003; Paek & Horvitz, 2000).

Recently, we have presented a statistical approach for the construction of a dialogue manager (Hurtado et al., 2006). Our dialogue manager is mainly based on the modelization of the sequences of the system and user dialogue acts and the introduction of a partition in the space of all the possible sequences of dialogue acts. This partition, which is defined taking into account the data supplied by the user throughout the dialogue, makes the estimation of a statistical model from the training data manageable. Using the information generated by the devices, captured by the sensors in the environment and provided by the context, the dialogue system manages the interaction with the user and provides services. As the dialogue between the system and the user is multimodal, the dialogue manager controls the execution flow from various input and output communication channels, e.g. speech and/or icon selection on a PDA screen. The modalities to be used may be determined by the user profile and the contextual information about the environment conditions. To enable this dialogue, the system employs predefined dialogue management strategies to decide what to do after processing the information obtained from the user, the environment and the context. Typical actions are: provide information to the user extracted from a database query, check or change the status of devices in the environment, confirm data, and ask for missing data necessary to perform a specific action. Information can be missing in a dialogue system either because a data item has not been provided by the user, or due to processing errors. For instance, a processing error can be observed in the dialogue shown above, as the data item 'in the evening' provided by the user in turn (5) was not understood by the system.

In order to enhance the system performance and provide a friendly interaction, the dialogue management strategies must be adaptive to user characteristics such as knowledge, expertise, preferences and needs, which are stored in the user profile (Forbes-Riley & Litman, 2006; Hassel & Hagen, 2005). Specifically, the adaptation to special needs is receiving much attention in terms of how to make systems usable by handicapped people (Heim et al., 2006), children (Bayeh et al., 2004) and the elderly (Langner & Black, 2005). The DM can apply different strategies to deal with possible errors. It can use implicit confirmations, but this strategy is only effective with highly reliable data, because implicit confirmation of erroneous data generates confusion in the user. It can use explicit confirmations on all the main items of the task. However, this kind of dialogue system strategy leads to very long dialogues because the system uses many turns in explicit confirmations. To avoid this drawback, the dialogue manager can be designed so that it makes use of explicit confirmations only when relevant uncertainties in the recognition and/or understanding processes are detected. Therefore, it would be desirable to provide the dialogue manager with information about what parts of the user utterance have been clearly recognized and understood and what parts have not. From this information, the dialogue manager could direct its strategy to confirming or asking only about the uncertain parts of the user utterance. The need to detect errors or data with low reliability makes the association of confidence scores to each component of the decoding (acoustic, lexical or semantic) relevant.

In the literature, several approaches have been proposed to detect and handle the errors generated in the recognition and understanding processes of the input utterances. In the case of the recognition process, supplying several sentence hypotheses (Nbest, graphs of words) instead of a single hypothesis, or associating confidence scores with words, allows the following modules of the dialogue system to work with some alternatives and to take the reliability of words into account (López & Mateo, 2005; Sama Rojo et al., 2005; Torres et al., 2005). Although the confidence scores associated to words in the recognition process are the most frequently used measure of the reliability of the information provided to the dialogue manager, it is also interesting to use confidence scores obtained in the understanding process. In (Bouwman et al., 1999; Sturm et al., 1999), the use of confidence measures associated to concepts or attributes in the understanding process has proven to be appropriate to guide the implicit and explicit confirmation strategies in the ARISE information system. There are also other similar contributions such as (Córdoba et al., 2001). The definition of two confidence measures in the understanding module is proposed in (García et al., 2003). One of them is related to the association of words and concepts, and the other one is related to the probability of sequences of concepts. In these cases, the understanding module provides the dialogue manager with the semantic representation (generally one frame or several frames) associated to the user input together with its confidence scores. To implement the user-system interaction by means of a multimodal dialogue, the system developers must design interaction strategies to control the way in which user and system interact with each other employing dialogue turns. For example, one issue to decide is whether the user must provide just the data items requested by the system, and in the order specified by the system, or alternatively, he is allowed to provide the data in the order that he wishes and be over-informative (provide data not requested by the system).

Another important issue to decide is who (user or system) has the initiative in the dialogue. In accordance with this criterion, three types of dialogue strategies are often distinguished in the literature: user-directed, system-directed and mixed. When the former strategy is used, the user has always the initiative in the dialogue, and the system just responds his queries and orders. When the system-directed strategy is used, the system has always the initiative in the dialogue, and the user just answers its queries. Finally, when the mixed interaction strategy is used, both the user and the system can take the initiative in the dialogue, consequently influencing the conversation flow. The advantage is that the system can guide the user in the tasks that he can perform. Moreover, the user can take the initiative and be over-informative.

After the analysis of the input information captured from the user and obtained from the environment, the dialogue system must generate a response for the user taking into account the context information and the user profile. To do this, it can employ the output frames. These frames are analysed by a module of the system that carries out the so-called fission of multimodal information, i.e., a decision about the devices and interaction modalities to be used for generating the response. For example, the system can generate a multimodal response using synthesised speech, sounds, graphical objects and an animated agent in a PDA screen. The use of several modalities to generate system responses allows a friendlier interaction that better adapts to the user preferences and needs. Moreover, it enables the user to create a proper mental model of the performance of the system and the dialogue status, which facilitates the interaction and decreases the number of interaction errors. The response generation may involve as well to access the middleware to change the status of a device in the environment, e.g. to switch on a specific lamp in the living room (Montoro et al., 2004).

2.5 Evaluation of context-aware systems

Evaluation is a central piece of software engineering. Evaluations methodologies allow researchers to assess the quality of the findings and to identify advantages and disadvantages of a system. The goal in evaluation of conventional systems is to proof that a system is more efficient. Normally, variables associated with efficiency are the time to complete a given task or the number of errors that have been made while fulfilling the task. However, in AmI when a system augments an environment enabling a user to do new things the metric is not straight forward anymore. So, it is important before evaluating a context-aware system to figure out what is the evaluation goal. Context-aware services must dynamically adapt to the needs of the user and to the current physical, social and task context in which those needs are formed. Developing an effective context-aware adaptive service therefore requires extensive user-centred design and evaluation as the proposed adaptive functionality for the service needs to evolve.

Since a few years, there is a growing interest in understanding specific evaluation problems that arise from context-aware systems (Dey et al., 2002; Scholtz, 2001; Scholtz & Consolvo, 2004). And (Schmidt, 2002) propose a taxonomy where we can find several methods for evaluating context aware systems. They are:

1. **Pre-implementation Evaluation:** (Dahlback et al., 1993; Salber & Coutaz, 1993) In many cases a method known as "Wizard of Oz" is used to predict the behaviour of the system (Dahlback et al., 1993; Salber & Coutaz, 1993). In this method, a human mimics the computer's behaviour to save implementation time. Humans are used to mimic or simulate tasks in which they are better than a computer, for instance, the prediction of behaviour time. If the system performance or the user experience is not as expected this could be an indication that it is useful to rethink the concept.

2. **Sub-system Evaluation:** (Schmidt et al., 1999a) Evaluation of sub-systems is the proposal here, and it can be practical to evaluate parts of the system by separated. Evaluation techniques like simulation and test runs for the case of network protocols; or usability test for new user interface concepts; or data collection and recognition performance analysis for context acquisition; or prototyping to proof feasibility of system design; are proposed, so the correct metric can be used to compare in each field. But there is a need to evaluate complete systems and this method do not provide it.

3. **Overall System Evaluation:** (Schmidt, 2002) This is not an easy task and it is classified by the author into four approaches:

   - Single domain focus: This evaluation offers a means to provide proof of advances made and also helps to identify new issues in this specific domain. Using this method the system is evaluated from the view of a single domain.
   - System Feasibility:This method focused on building a demonstrator showing the implementation can be done, so it can provide a proof of concept for a particular system.
   - Prototyping: It makes possible to experience the system as it is intended and can therefore give more insight on different usability or implementation issues.
   - Living Lab: It refers to the same prototype but having them in everyday use, as people work in labs it is better to test prototypes in the living laboratory style environment.

But context-aware systems are designed to provide users with services but where the main point arises in the potentiality of context. This kind of systems are designed to help users in a certain situation and provided information that is useful for a particular task (Schmidt, 2002). Evaluation procedure becomes then a difficult problem to face when dealing with this kind of systems. If we assume that the main purpose of context-aware system is to provide a user with information according to the contextual information, there are several problems the user currently has in accessing this information:

1. **Distribution of the information:** It concerns the manner in which the information is requested and disseminated to a user. Currently the user has to realize a need for information and make this request to the computer using whatever interactions are available in the user interface.

2. **Volume of information:** It is certainly an issue; there is an increasing amount of information available and it is growing at a rapid rate. It is extremely difficult for the user to sort through all and ascertain the information that is relevant to the situation.

3. **Level of information:** It is also a problem. The user is faced with having to assimilate much information and interpret it to answer a higher level need.

4. **User Mobility:** As users move between environments, computing resources change. Data has to be move so that it is accessible; applications have to be restarted in a different computing environment. Much of this responsibility falls on the end user currently and is another distraction that the user must deal with in order to access information.

There are several approaches to evaluate context-aware systems. Groups such as the Future Computing Environments Group at Georgia Institute of Technology working on the "Aware Home" (Abowd et al., 2000) and Tatsuya Yamazaki of National Institute of Information and Communications Technology, Japan working on the "Ubiquitous Home" (Yamazaki, 2005) have completed real-life test home environments for accurate simulation of the home environment. Both groups aim to perfectly emulate a real domestic environment and intend to have test-subjects spend significant periods of time in these simulated home environments carrying out domestic activities. However, such live usage test beds are expensive and difficult to reconfigure to emulate a wide range of different contexts.

Kerttula and Tokkonen (Kerttula & Tokkonen, 2001) have identified "the total user experience" as an area of concern and aim achieve it through early product and system simulations. This idea moves away from testing in isolation and moves towards a simulation where services are tested in parallel and valued over longer periods of time. This approach uses accurate simulation/prototyping of services focussing on features such as the user interface, audio properties and product behaviour, but not including the user's surrounding physical environment.

Huebscher and McCann (Huebscher & McCann, 2004) aim to allow initial testing of context-aware applications without requiring a physical deployment. However Huebscher and McCann are working to simulate sensor data e.g. temperature, humidity or location, from a description of context or a simulation model of contexts. This in turn will be used to test the context-logic of a context-aware application.

In the past, virtual reality simulation of pervasive computing environments has been used in a small number of research efforts, specifically QuakeSim (Bylund & Espinoza, 2002) and HP Lab's UbiWise (Barton & Vijayaraghavan, 2002). These have demonstrated that 3D virtual reality computer game engines potentially provide a cost effective platform for simulating pervasive computing environments with sufficient realism to accurately test human interaction with pervasive computing software systems.

Also, Shirehjini and Klar have been developing 3DSim (Nazari Shirehjini & Klar, 2005), a 3D tool for rapidly prototyping Ambient Intelligence building blocks e.g. situation-recognition, goal-based interaction. 3DSim aids the development of human-ambient-interaction systems such as PDA based control systems, adaptive user interfaces, multimedia output coordination or goal-based interaction systems. During a simulation, sensor data is derived from a 2D GUI and gesture elements which are the result of an avatar can pointing at devices.

The team at GIST U-VR Lab, S. Korea have been working on creating a unified context model and a method for the integration of contexts for unified context-aware applications. To loosen the coupling between services and context, they have developed a unified context that represents user-centric contextual information in terms of 5W1H (Who, What, Where, When, How and Why) (Jang et al., 2005). To demonstrate user-centric integration of contexts for a unified context-aware application model (the ubi-UCAM), they created a simple 3D simulated environment (Oh & Woo, 2005). By using the simulator they were able to test the effectiveness of the Context Integrator when there were multiple users working with the service simultaneously. The simulated environment allowed them to assess the capabilities of their Context Integrator before bringing it into a real world situation (ubiHome).

Other examples of context-aware system evaluations are in AmI where sensory information is used to determine the context of a situation and the application modifies its behaviour accordingly. The behaviour could be an action performed automatically or information that is modified based on the context of the user. Other researchers in augmented cognition and affective computing are attempting to understand the cognitive state of the user and modify the delivery of information based on what the user can currently handle. Evaluations of such systems cannot be done solely through empirical methods as there are too many situations to test. Simulation could be used as a supplement though human judgement of the appropriateness of the modified behaviour needs to be the ultimate metric (Bylund & Espinoza, 2002).

One possibility is to classify modified behaviours according to a severity scale and test representatives of the various classes. Possible measures include: time savings by having the action automatically performed, reduction in user cognitive load by having the action automatically performed, the time needed to "undo" an inappropriate action or the severity of not "undoing" the action, the percentage of appropriate actions in a given experiment.

Mobile computing and ubiquitous computing deliver information to users who are conducting primary tasks. The information delivery is beneficial but is a secondary task. A possible measure here is the time taken from the primary task to obtain this information. (Smailagic et al., 2001) suggest using levels of distraction as measures of a secondary activity. They propose 4 classes of distractions: snap, pause, tangent and extended. Snap activities take a few seconds and does not interrupt the primary activity. Pause activities take several minutes and will require the user to stop a primary activity but should not cause severe problems in restarting the primary task. Tangent and extended activities cause the user to switch from the primary task and concentrate on the secondary task. Others have also suggested various metrics for evaluation.

- Jameson (Jameson et al., 2003) proposes five usability challenges for adaptive interfaces (i.e., systems that learn from the user's behaviour and react accordingly): (1) predictability and transparency, (2) controllability, (3) unobtrusiveness, (4) privacy, and (5) breadth of experience.

- Friedman and Kahn (Friedman & Kahn Jr, 2002) suggest 12 key human values with ethical import: (1) human welfare, (2) ownership and property, (3) freedom from bias, (4) privacy, (5) universal usability, (6) trust, (7) autonomy, (8) informed consent, (9) accountability, (10) identity, (11) calmness, and (12) environmental sustainability.

- Bellotti et al. (Bellotti et al., 2002) suggest five interaction challenges for designers and researchers of sensing systems: (1) address—"directing communication to a system," (2) attention—"establishing that the system is attending," (3) action—"defining what is to be done with the system," (4) alignment—"monitoring system response," and (5) accident—"avoiding or recovering from errors or misunderstandings."

These challenges could also be used for evaluation of context-aware systems but these pertain only to interaction.

In AmI environments, user context and user preferences become essential aspects when deciding, which of the available services are of most interest to the user in a given situation (Rasch et al., 2011). The user context covers aspects such as user location, current time, environmental information and the users status (Dey & Abowd, 2000). These changing contexts are a challenge to the service discovery mechanisms. There are two ways of service discovery: explicit (where the the user makes a explicit request to the system) and implicit (without user's request). In AmI environments, most of service discover request are implicit because of the same nature of this kind of environment. We are now well into the second decade of progress in pervasive computing, and as a community continue to see new middleware and programming models being proposed and prototyped.

However, we lack higher-level evaluations that can increase our overall understanding and help to advance in context-aware systems researches. And particularly focusing on evaluating context aware systems in AmI environments, we currently lack a systematic user-centred evaluation methodology for interactive systems.

Without a solid foundation for evaluating the progress in pervasive systems, it is impossible to assess where we are today and appropriate directions for future research. There remains a need for some type of quantitative measurement, or a collection of common set scenarios to assess and compare context-aware systems. One of the difficulties in evaluating context-aware systems is their openness and diversity. There are extremely diverse domain of applications and the corresponding huge differences in users' needs and expectations, so the challenge, therefore, is to find a common set of scenarios that can serve not only as a demonstration of the true potential of pervasive computing technology, but also as a model of research for the research community.

2.5.1 Evaluation metrics: What to measure?

Evaluations for context-aware systems can not be conducted in the same manner evaluation is understood for other software systems where the concept of large corpus data, the establishment of ground truth and the metrics of precision and recall are used. Evaluation for changeable systems like context-aware needs to be conducted to assess the impact of the users. The heterogeneity, dynamicity, and heavily context-dependent behaviors of context aware systems require new approaches of performance evaluation. Normally, apart from the simulation techniques, real-world evaluation is conducted as field studies and relies on collecting data from observation about the usability of the software in the context of use. But there is a need of assessing usability of the software and measure the utility of the context-aware system. A three level approach to evaluation is proposed by (Scholtz & Consolvo, 2004):

- evaluation of the science;

- evaluation of the component technology, and
- evaluation of the impact.

2.5.1.1 Impact metrics

Potential system impact metrics include: (a) Trust in the system, (b) Shift in user time, (c) Increased quality of product, (d) Increased confidence in product. Measurement of these attributes will encompass both quantitative and qualitative data. Trust in the system can be measured by the amount of system suggestions that information analysts further explore and by user ratings and comments. Trust in the system may also be a measure of the understanding the user has of how the system actually works. In that respect assessing the user's mental model of the system is appropriate (Carroll et al., 1987; Norman, 1988).

Product quality can be obtained through judgements of subject matter experts. Better coverage of data may result in an increased confidence in the analyst's recommendations. Increased confidence in the product can be measured by users ratings. Other overall metrics might include process improvements. To measure this, a cognitive task analysis can be conducted to provide insight about obstacles in the process. Noting where and what those obstacles are, we can use our baseline data to determine if these obstacles are removed once research tools are inserted. While overall process metrics are the long-term goal, we also want to be able to provide component based metrics. That is, if we find that the time shift now allows users to concentrate on higher value tasks, we would like to be able to attribute this shift to the component(s) responsible.

2.5.1.2 Component level metrics

Component metrics are, of course, specific to the particular functionality of the component. To provide some examples of possible metrics, we go back to some current research efforts. In particular, we consider potential metrics for question and answer dialogues, user modelling, hypotheses generation, prior and tacit knowledge. Question and answer systems currently being developed in the research community go beyond simple Question and Answer systems. These systems might help a user construct a travel itinerary, with flight reservations, hotel bookings and car rentals. Evaluation efforts for dialogue systems have used user satisfaction as the impact metric. Component metrics for question and answer dialogues might include: (i) Completeness of answer; (ii) Accuracy of answer; (iii) Effort required on part of user engaging in dialogue.

Recent evaluations have been done of question and answer dialogues in speech research and in the information retrieval world (Walker et al., 2001). If the question is reasonably simplistic, a template can be constructed of the different variables that need to be filled in to constitute an answer. Upon completion of the dialogue, the percentage of variables filled in might be used as a metric. While this is a reasonable component metric, the dialogue may be of no value to the user if the answer is not complete. However, if we compared components based on the percentage of the template filled in, we might predict that components achieving a higher percentage of success will contribute more to impact metrics than components with a low completion rate. Metrics used in these evaluations were:

- Dialogue efficiency metrics such as total elapsed time, time on task, user turns, system turns, and turns on task.
- Dialogue quality metrics including sentence error rate, word error rate.
- Task success metric – user perception of success.

The Paradise framework (Bonneau-Maynard et al., 2000; Hirschman, 2000; Hirschman et al., 1993; Price et al., 1992) was used in these evaluations to integrate the above metrics to predict the ultimate metric of user satisfaction. Researchers in user modelling expect that by understanding more of what the users are doing, intelligent systems will be able to provide appropriate help. Metrics for consideration here include time, accuracy, benefit:

- time user spends explicitly entering data/ user critique of model/ time for model to adjust to changes in interest
- comparison with current system and what information is delivered to user
- time saved/time spent (comparison)
- number of helpful interactions/total number of interactions

Components scoring well on these metrics will likely contribute to the time shift measure proposed for an impact metric. Researchers looking at hypotheses generation intend to develop systems that can suggest hypotheses to the analyst that may have been overlooked. Assuming that we have knowledge of the data collection being used, we could use human assessors to judge the relevance of generated hypotheses. Another measure might be the percentage of hypotheses generated by the system that cause the analyst to do some further exploration. The generation of a large number of hypotheses for the information analyst to consider might increase confidence in the end report.

However, having to sort through a large number of hypotheses might not produce the time shift distribution that we desire. This illustrates the necessity of looking at the mapping between component and impact metrics. Evaluation components that make use of prior and tacit knowledge are more problematic. A subject matter expert in a particular area would be pleased to have information filtered out that he or she already knows. However, intelligence information is dynamic. Just because a country's economy was stable several months ago does not mean it is currently. Therefore temporal aspects of the prior knowledge have to be considered as well. One measure could be the percentage of previously known information/ all the information returned with the appropriate time intervals factored in.

2.5.1.3 Scientific metrics

Scientific metrics are still important and need to be the first evaluation applied. Consider the same topics discussed in component metrics: question and answer dialogues, user modelling, hypothesis generation, prior and tacit knowledge. Scientific metrics for question and answer dialogues might include:

- Recognition and handling of ambiguous questions or statements

- Context tracking for threads of dialogue
- Ability to engage in mixed initiative dialogue with end user
- Percentage of simple questions recognized and accurately answered

The domains for the discourse would have to be considered as well as any constraints on the vocabulary of the end user. Scientific measures for user modelling might include:

- Ability to construct user model based on recognition of user activities
- Accuracy of user model
- Predictability of user model

The various types of user activities that can be recognized and used by the system would be a consideration in this scientific evaluation. Algorithms that generate and track hypotheses would have to demonstrate the ability to:

- Generate hypotheses from data
- Recognize user generated hypotheses
- Assemble evidence or contradictory information from data for multiple hypotheses

These capabilities should be evaluated using data collections of various sizes and with different signal to noise ratios. The TREC novelty track might be a reasonable starting point for scientific evaluation for prior and tacit knowledge. Given a number of documents, can algorithms find only the new information in successive documents.

2.5.1.4 Mapping metrics

Intelligent information systems need to demonstrate good science first. Scientific evaluations are controlled, laboratory type evaluations. Different algorithms can be evaluated and compared. Items such as number of domains, vocabulary, size of data collections, signal to noise ratio can be manipulated to test the algorithms under different conditions. Before integrating these algorithms directly into a system for evaluation we recommend a component level evaluation that uses user-centric metrics. These evaluations may or may not be done empirically. The metrics used in these evaluations will also be used in the final impact evaluations.

For example, consider hypothesis generation and tracking algorithms. Suppose that a scientific evaluation showed that the algorithm was capable of recognizing hypotheses that an information analyst was investigating and could find alternative hypotheses in large, noisy data collections. The component metrics showed that a high percentage of the hypotheses brought to the attention of the analyst were used for further exploration. An impact evaluation showed that the analyst confidence in the final report was much higher than usual. If all other parts of the analysts environment were constant, we could attribute this increase in confidence to the ability to explore more alternative hypotheses.

The three types of metrics for evaluation (scientific, component level, and impact) apply to AmI as well. Consider a context-aware tour-guide system. This system recognizes where the user is, gives the user information on a particular attraction, and possibly recommends another attraction that the user would like to visit. The user would also be able to inquire about restaurants and stores in the vicinity. At the scientific level, metrics would address the recognition of a number of attractions and the ability to deliver the information corresponding to the attraction. Evaluation would also look at the ability to recognize and appropriately handle user inquiries. At the component level, we would measure the utility of the information delivered on attractions. We would also evaluate the accuracy and completeness of the information delivered with respect to user inquiries. At the impact level, a possible metric would be the satisfaction rating of the user of their visit to this particular city.

2.5.2 Metrics for AmI

For evaluating the performance of various context-aware systems we utilize a three-facet approach. First, we use a taxonomy of pervasive computing systems based on our survey of proposed and prototyped systems and research projects (Abdualrazak et al., 2010). Second, we create a set of case scenarios (Griol et al., 2010; Sánchez-Pi et al., 2008a,b, 2007; Sanchez-Pi et al., 2011; Sánchez-Pi et al., 2010; Sánchez-Pi & Molina, 2009a,b, 2010a,b; Sánchez-Pi & Molina, 2009, 2010) which serves as a checklist of goals and functionalities for system designers to consider during both design and implementation stages. Third, we identify critical parameters for evaluating context-aware systems and a list of parameters allowing to decide what and how to evaluate each case scenario.

Taking all these into account we present in Chapter 3 a methodology for context-aware systems in AmI environments based on 2.1. The taxonomy was constructed using a bottom-up approach and includes: architecture, modularity, geographic span, purpose and integration criteria, to define categories and key parameters to measure each one.

Numerous design methods have been suggested to assist in the development process, and many are applicable to pervasive systems. But the most widely known are user-centered design and we will focus on it. User-centered design is a set of techniques and processes that enable developers to focus on users during the design process. With this design methodology, users would get involved in the development process from the very beginning, depending on their skills and experience, and their interaction is facilitated by a domain expert. The intensity of their involvement varies throughout various stages during the research and product development cycle. So in the evaluation process we will focus the purpose criteria: assurance, assistive, return of investment, experience enhancement and exploration.

CRITERIA	CATEGORY	KEY ASPECT
Architecture (Infrastructure)	Centralized	Resource Usage
	Distributed	Deployment
		Safety
	Grid	Resource Usage
		Safety
	Mobile (Infrastructure)	Resource Usage
		Invisibility
	Mobile (AdHoc)	Speed & Efficiency
		Resource Usage
		Safety
		Deployment
		Compatibility
		Invisibility
Architecture (Design)	Application based architecture	Modularity
		Software dynamics
		Design
	Component Oriented Architecture	Modularity
		Software dynamics
		Management
		Design
	Service Oriented Architecture	Modularity
		Compatibility
		Software dynamics
		Management
		Design
	Agent Oriented Architecture	Management
		Design
Application purpose	Assurance	Safety
		Sentience
	Assistive	Usability
		Safety
		Invisibility
	Return on Investment	Speed & Efficiency
	Experience enhancement	Sentince
		Usability
	Exploration	Sentience
		Deployment
Autonomicity	Static	Speed & Efficiency
		Safety
	Dynamic	Sentience
		Usability
	Re-Programable	Programmability
		Deployment
		Compatibility
	Re-Configurable	Usability
Integration	AdHoc Integration	Method
	Universal Interface	Method
	Plug-In	Method
Interaction	Human to Machine	Human Capabilities
		Preferences
Intelligence	Machine to Machine	Interoperability
		Context awareness
Service Availability	Any where - Any time	Discovery
		Location
		Adaptation
		Availability
		Mobility

Figure 2.1: Taxonomy of Pervasive Computing Sytems. Taken from (Abdualrazak et al., 2010)

3

Intelligent Systems Development for the Provisioning of Services in AmI

From a technological perspective, challenges are present from many perspectives: the real availability of the foreseen technology, its integration in a working system, the interaction with the emerging environment, and the acceptability of the resulting services by the potential users. There are several concepts are considered as a basis for the emergence of an intelligent environment: embedded intelligence, middleware and distributed systems, IP mobile and wireless, multi-domain network management, converging core and access networks, micro and opto-electronics, trust and confidence enabling tools, cross-media content, multi-modal and adaptive interfaces, multi-lingual dialogue mode.

The following requirements are considered crucial for the technological developments. The first is that the hardware must be very unobtrusive. Many technologies are conceived as hand-held or wearable, taking advantage of the intelligence embedded in the environment. This means being lightweight, but also available. People will have with them everything they need to perform any tasks. The necessary intelligence can be in the environment and the only limiting factor can be the bandwidth. This hardware must be integrated in a seamless mobile/fixed web-based communications infrastructure. Complex heterogeneous mobile and fixed networks need to interoperate and to be dynamically reconfigurable. This will allow the deployment of dynamic and massively distributed device networks. The AmI landscape is a world in which there are almost uncountable interoperating devices (wire, wireless, mobile, and fixed).

Interactions must be through natural human interfaces and with systems that are intuitive in use. This will need artificial intelligence techniques as the basis for intelligent agents and human to machine interactions, which are supposed to be multimodal (multi-user, multilingual, multi-channel and multipurpose), context sensitive and able to filter information. Finally, a very important design requirement is dependability and security. The AmI world must be safe, dependable and secure, considering all physical and psychological threats that the technologies might imply.

In this chapter we explain our approaches to the provisioning of services in AmI environments. First, we detailed a centralized development for context aware systems in AmI scenarios. Afterwards, a distributed development based on intelligent agents is presented as a solution for systems in the scope of AmI. Multimodal interaction paradigm is also introduced by means of a dialogue system to provide with an intuitive way to access to the information in this systems. Finally the evaluation methodology for AmI systems is addressed.

3.1 Centralized system for AmI environment

Context-aware systems can be implemented in several ways. It depends on conditions like e.g. the amount of users, the available resources of the used devices (PCs or small and handheld devices) or the facility of the system. According to (Guanling Chen & Kotz, 2004) there are three different approaches based on the above considerations. The approach of Direct Sensor Access is often used in devices with sensors locally built although it is not suitable for distributed systems due to its natural direct access. The Middleware-based design uses methods of encapsulation and introduces a layered architecture to context-aware systems. The other logical approach is to permit multiple client access to remote data sources in a Context Server.

There are several approaches in context-aware commercial applications with different implementations. RCSM (Yaua & Karim, 2004) and CORTEX (Biegel & Cahill, 2004) are context-sensitive middleware frameworks which build context-aware applications on the creation and composition of sentient object. There are others like Entree (Burke et al., 1996) which uses a knowledge base and case-based reasoning to recommend restaurant or for instance Cyberguide (Abowd et al., 1997) project which provides user with context-aware information about the projects performed at GVU center in Atlanta. With TV remote controllers throughout the building to detect users locations and provide them with a map that highlights the project demo available in the neighbouring area of the user.

After reviewing some of the context-aware solutions, it is clear that these applications must be able to adapt to context changes in order to provide users with quality of service. In other words, context-aware systems are expected to utilize contextual information to adapt their behavior based on a predefined set of rules. Such rules are mostly, monitored by a system that dynamically adapts its operations based on this information. Appear (Appear Networks, 2008) is a commercial platform and a good solution for context management, that is why we selected it to model and test our prototypes.

But there is an increasing need to use KBS to represent complex problems like context-aware systems. As a result of the increasing development of these kinds of systems and applications, more knowledge-based systems (KBS) are also being developed for use in areas where context-aware system failures can be costly because of losses in services, property, etc. Their architecture and representation formalisms are the groundwork for today's systems. The knowledge is usually derived from expertise in specific areas and is validated according to a different methodology than is used in conventional systems because the knowledge is symbolic. Unlike conventional systems, in KBS, domain knowledge is represented explicitly, it is separated from the general reasoning algorithm used for processing it and the internal operation is based on non-deterministic processing, that is why they are adequate for representing context-aware systems. There are several methodologies as stated in Chapter 2 but we have chosen CKADS to model our problem because it covers the main points of the development of a KBS from the very start of analysis (for identifying the problem and establishing the feasibility of a KBS-based solution) to the implementation of the project.

In this section the context aware centralized system is presented. First we describe the Appear platform, its architecture and specifications, it context representation and reasoning engine. Afterwards, we explain how we used CommonKADS(CKADS) methodology to acquire and reason knowledge in a knowledge based system (KBS). Later we precise the Appear context configuration using CKADS. Some of these explanations are supported on an airport domain, which will be detailed in Section 4.1.

3.1.1 Appear platform

Appear is an application providing a solution for a wireless network. This solution enables the distribution of location-based applications to users in the proximity of predefined interest points. All Appear needs is a IP-based wireless network and a Java-enabled wireless device. Appear uses a platform-independent external positioning engine in order to locate devices and calculate their position.

3.1.1.1 Architecture

The Appear platform consists of two parts: the Appear Context Engine, which is the core of the system, and the Appear Client, which is installed in the device. Applications distributed by the Appear Context Engine are installed and executed locally in these wireless devices. The architecture of the Appear Context Engine is modular and separates the system responsibilities into server, one or more proxies, and a client. See Figure 3.1. The Appear Context Server is part of the network management. It manages the applications distributed by the platform and the connections to one or more publishers, or proxies or positioning engines.

The Appear Context Proxy is typically installed inside but can also be located outside the network. Its main function is to eliminate unnecessary traffic between the local network and the main server since the bandwidth is usually limited. It keeps a cache of the active user sessions and the most accessed services. When a user requests to download a service, the proxy checks if it can be served from the cache instead of from the main server in order to carry out its main function. When a wireless device enters the network, it immediately connects with a local proxy, which evaluates the position of the client device and initiates a remote connection with the server. Once the client is in contact with the server, they negotiate the set of applications that the user can access depending on his or her physical position. The Appear solution consists then of the Appear Context Engine and its modules: Device Management Module, Push Module and the Synchronization Module. The three modules collaborate to implement a dynamic management system that allows the administrator to control the capability of each device connected to the wireless network.

Provisioning module The Push or Provisioning Module manages the automatic distribution of applications and content to hand-held devices. It pushes services on these devices using client-side intelligence when it is necessary to install, configure and delete user services. There are six steps in the Context Aware Provisioning:

- Device Detection: The network discovers the client and provides it with the required configuration.

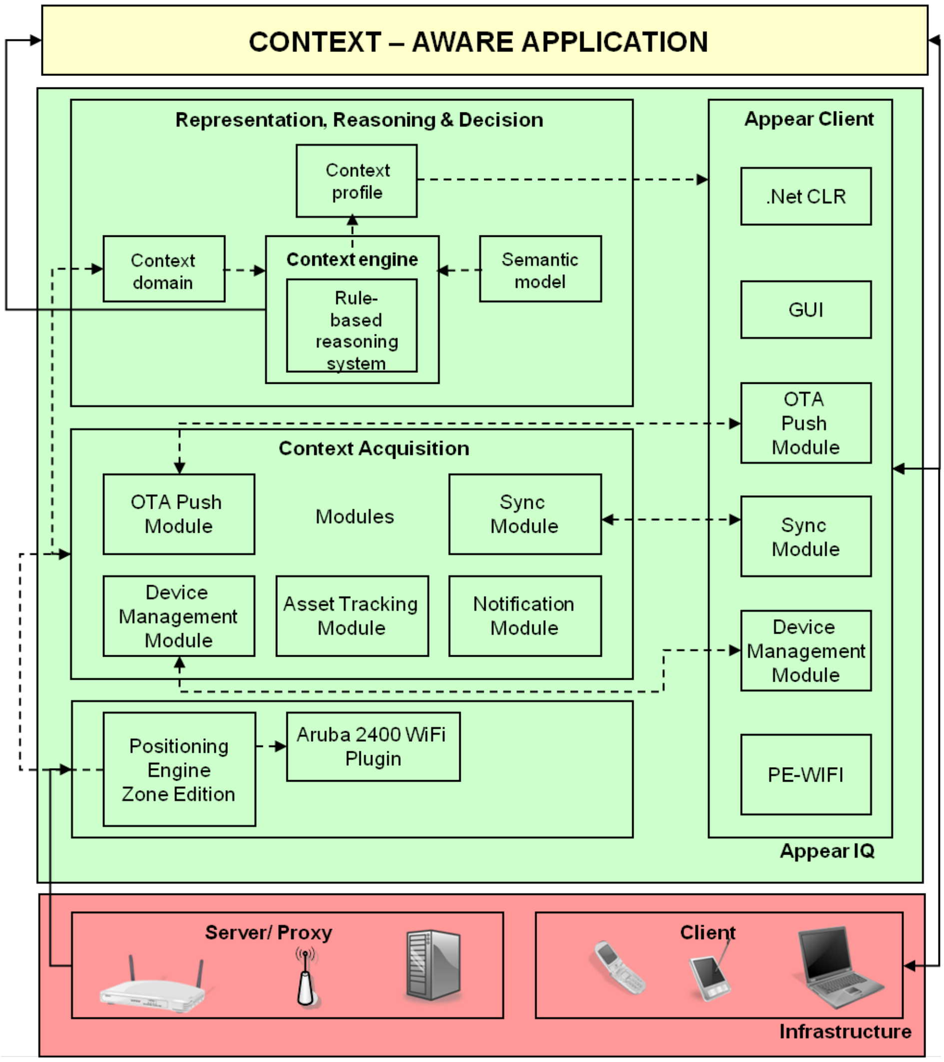


Figure 3.1: Centralized architecture using Appear platform.

- Service Discovery: The Context Profile is made up by the Context Engine and it then compiles an offering for the client. The offering is shown as icons on the handheld device using a user interface. The client then decides which service to pull by clicking on the respective icon.

- Download: The download of the necessary resources is the first step after the service discovery.

- Install: Once the resource is available on the device it will be installed.

- Execute Service: Once the service has been launched, the application is execute.

- Discard Service: A service will be discarded when a user leaves the network or if the context condition for a service changes.

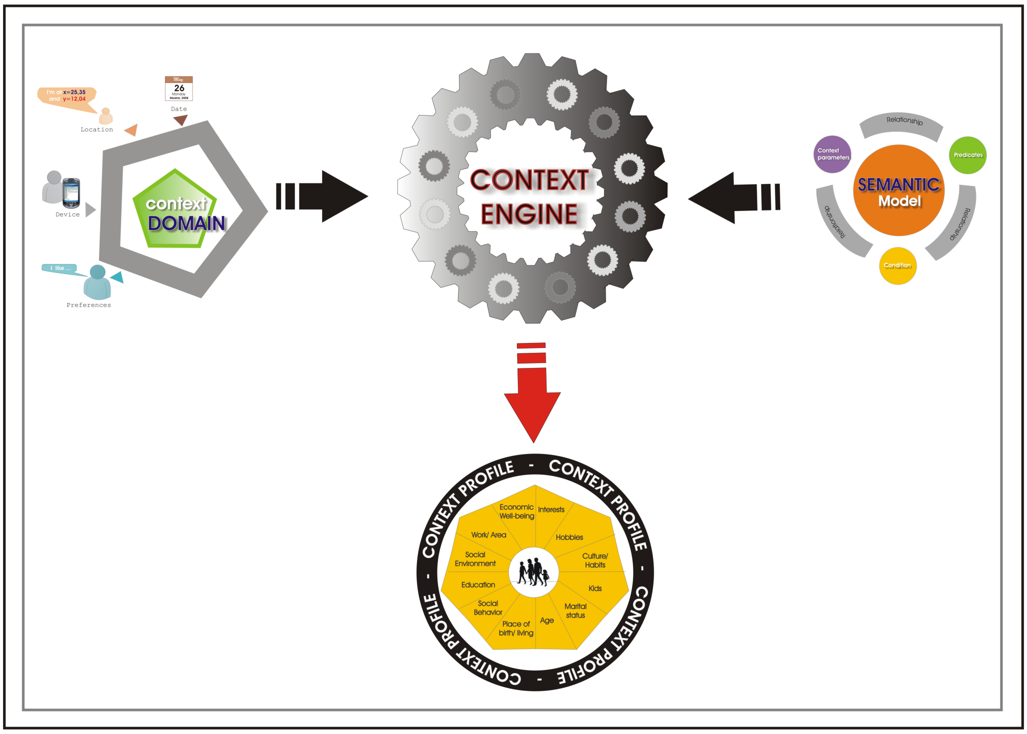


Figure 3.2: Interaction components Appear.

Device Management Module The Device Management Module provides management tools to deploy control and maintain the set of mobile devices. The context-aware actions in the client side are:

- The configuration elements that describe the specific steps to be taken by the client. They are initially installed together with the client and then it is updated using the Synchronization Module.
- Context conditions: An associated condition to the current context is applied to determine if the action is applicable. It is made by the rule-engine of the client.
- Mirroring: It is a mechanism by which the client monitors file updates on the device. These updates are replicated to a secondary device as a storage card or a remote host using FTP/HTTP.

Synchronization Module The Synchronization Module manages file-based information between corporate systems and the mobile handheld devices. The Device Management is continuously updated with up-to-date versions of the configuration files.

- The Synchronization Module compiles contextual data to gain an understanding of the user's informational needs.
- Available data is filtered against the user's context to determine what information should be the most relevant.

- The Module automates synchronization, detecting files that have change and synchronizing them. It is a dynamic synchronization of the profile based on User and Role, Location, Time, Device Status and Connectivity. All of these modules are made context aware using the Appear Context Engine. See Figure 3.2.

3.1.1.2 Context representation

In Appear, the Appear Context Engine gathers the context of user data and builds a model based on the needs of the end user. It implements a rules engine, which determines which service is available to whom, and when and where it should be available. Services are filtered against a profile, and when it is decided that data are relevant, the information is proactively pushed to the device. As mentioned, the Appear Context Engine gathers all the context information about the device and produces a context profile for that device. The main components of this model are Context Domain, Context Engine, Context Profile and Semantic Model.

- Context Domain is a set of context values the system can monitor. In the context domain all values are given without any internal relationship. It is fed with context parameters that measure real-world attributes that are transformed into context values. Context parameters include physical location, device type, user role, date/time, temperature, available battery.
- The Context engine is the one that matches the context domain onto the semantic model and the result of it is the Context profile. See Figure 3.2
- Context profile components are classified in different categories: Behaviour (social behaviour), Personality (habits, interests), Demographic information (age, place of birth, marital status, kids), Economic information (economic or well-being, work or area) and Social indicators (social environment, hobbies). Context profile contains all context values and values of predicates based on the context values.
- Semantic model is the Administrator model of the relationship between different context parameters and how these should be organized, using context predicates.

There are several lowest level objects, called context parameters managed by the Context Engine and they should be taken into account when deciding what to do in the system. To get into a more abstract level Appear creates more complex predicates combining and constraining the values of these context parameters and other context predicates. A predicate is a set of simple rules or conditions combined to create a more complex predicate. A predicate is constructed out of one or more conditions and an evaluation criterion that can be “match all” or “match any”. Match all requires that all conditions are evaluated as true. Match any requires that at least one condition is evaluated as true. A condition can be a context parameter condition or a condition on another predicate. The context parameter condition is a constraint on a specific parameter string value. The keyword can be evaluated against a condition (“is”, “is not”, “contains” and “does not contain”) and the expected value (i.e. zone airport=zone airport” is true). The predicate condition is a reference to another context predicate and the expected result is the evaluation of that predicate (i.e. if "zone airport then news service").

Context Profile components are classified into different categories: behaviour (social behaviour), personality (habits, interests, etc. . .), demographic information (age, place of birth, marital status, kids, etc. . .), economic information (economic/ well-being; work/area, etc.) and social indicators (social environment, hobbies, etc...). The Context Profile contains all context values and values of predicates based on the context values. Context information in the system is used throughout the entire service lifecycle. The rules engine filters and determines which service should be pushed to each user, when and where.

3.1.1.3 Scalability issues

Context- aware systems gather contextual information from many sources and process it to create the final representation available for a variety of clients. The context-aware system developed has a hierarchical architecture with the following layers: Web Services/Application Layer, Context Reasoning Layer and Sensor Layer The hierarchical architecture reflects the complex functionality of context-aware systems as shown in the following brief description of the functionality of particular layers:

- Sensor Layer. The lowest level of the location management architecture is the Sensor Layer which represents the variety of physical and logical location sensor agents producing sensor-specific location information.
- Context Reasoning Layer. This layer takes sensor-specific location information and other contextual information related to the user and transforms it into a standard format. This is where the reasoning about context takes place.
- Web services/ Application Layer. This layer interacts with the variety of users of the context aware system and therefore needs to address several issues including access rights to location information (who can access the information and to what degree of accuracy), privacy of location information (how the location information can be used) and security of interactions between users and the context-aware system.

The scalability issue of context-aware systems is important. For instance, there could be a great difference between the frequency of location updates from Aruba sensors and the frequency of location information requests from the users of the system. Moreover, once a location is determined at the Context Layer it is cached as a complete location description which will be valid until another update pertaining to that user. As a result, several requests for a single user's location can be served without significantly increasing the cost of generating the location from individual location fragments if location information updates are not very frequent.

3.1.2 KBS definition using CommonKADS (CKADS) methodology

The representation of knowledge in KBS is varied. For instance, the development of Semantic Web technology gives the information on the current Web a precise meaning and machine-interpretable form providing computers and people processing the same data with a common understanding of what the terms mean (Berners-Lee & Miller, 2002). The Semantic Web is also used in KBS development through ontologies that enable the construction of KBSs through reusable components across domains and tasks (Gomez-Perez & Benjamins, 1999). Ontologies are used to represent domain knowledge in knowledge-based programs. This is achieved using formal declarative representations of the domain knowledge; that is, sets of objects and their describable relationships (Gruber, 1993). Researchers in the area of knowledge modelling have started to realize the importance of ontologies in developing domain models since the underlying principle of modelling is to achieve agreed representations in a unified manner for the domains in which they are investigating. Ontologies can be described using Semantic Web languages, such as DAML+OIL, SHOE and RDF, and these descriptions are used to create the knowledge base of the KBS (Noy et al., 2001).

This way, the KBS developer can focus on domain knowledge representation instead of markup tags and correct syntax to build KBS faster. There are also several methodologies for the development of these systems. The best known is KADS, and this is the methodology that we will use in this section to formally represent our problem defined in Appear. KADS (Knowledge Acquisition and Document Structure) and its successor CommonKADS (Schreiber et al., 1999) (which is the de facto European standard) is a knowledge engineering methodology. CKADS is a structured approach to the development of KBS and, as such, is to be seen in contrast to unstructured approaches such as rapid prototyping. CKADS does not require a commitment to any specific implementation paradigm. According to KADS, the development of a KBS is to be seen as a modelling process, during which acquired knowledge is modelled at different levels of abstraction.

KADS identifies three levels of models:

- The process level identifies the tasks involved in the domain, the nature of data flows and stores, and the assignment of ownership of tasks and data stores to agents.
- The system level, the co-operation model describes in detail the interactions between the system and external agents, and how the internal agents interact. The co-operation model is used to separate the user task model from the system task model, and allows the knowledge wholly internal to the system to be readily identified.
- The expertise level corresponds to an expertise model. This divides the task of describing domain and expert knowledge and its use within the system into a number of supportive tasks. The layer-based framework for expertise consists of:
  - The domain layer is comprised of static or slowly changing knowledge describing concepts, relations, and structures in the domain.
  - The inference layer reformulates the domain layer in terms of the different types of inferences that can be made.
  - The task layer defines knowledge about how to apply the knowledge in these two layers to problem-solving activities in the domain.
  - The strategy layer concerns selecting, sequencing, planning and repairing tasks.

Within this framework knowledge engineering becomes a structured search for appropriate strategy, task, inference and domain models. The layers can be modelled in three steps: a) determine the static features or domain ontology, b) obtain the inferences that describe the dynamic side, and c) group the inferences sequentially to form tasks.

3.1.2.1 Domain layer

This layer represents the representative knowledge of the domain. It is here where the information and the static knowledge are described. Concepts in CKADS (Schreiber et al., 1999) are used to define object collections with similar characteristics. In the case of the airport domain, the concepts are identified as:

- Airport: This is the high-level concept representing the domain of discourse.
- Location: This includes the airport location, zones and the user location.
  - (x,y): The coordinates of the airport location, zones and the user position.
    - Zone: A segment of the airport area distinguished from adjacent zones by different characteristics. The coordinates. represents the zones in the domain.
      - Airport zone: This is the general zone. It contains the other zones. Every service available in this zone can be seen in the other zones.
      - Customs zone: This is where the customs authority is located in order to control the flow of goods.
      - Commercial zone: This is where restaurants and stores are located.
      - Offices zone: This is where the airline offices are located.
      - Check-in zone: This is where the check-in desks are located.
      - Jetway zone: This is zone leading to the aircraft. It contains the plan of the current flight.
- User: Every person who has a role in the domain.
  - Role: Role of each user in the domain
    - Pilot: Airline captain and pilots.
    - Passenger: Passenger.
    - FTO: Flight engineer
    - CBC: Cabin crew.
    - ASBG: Airline ground staff: boarding.
    - ASIC: Airline ground staff: incidents.
    - ASSV: Supervisor.
    - ASCH: Airline ground staff: Check-In.
- Offering: A class containing a set of services grouped by categories.
  - Category: A class containing a set of grouped services
  - Services: Applications or notifications
    - Flight Plan: A web service that delivers the flight plan of the current flight. It is available to pilots and FTO.

```
CONCEPT user ;
DESCRIPTION :
  "Every person who has
   a role in the domain"
ATTRIBUTES :
  Name: STRING;
  Address: STRING;
  Age: NATURAL;
  Role: value-role;
  …
AXIOMS
  …
END CONCEPT user;
```

```
VALUE-TYPE valor-role ;
  TYPE: ORDINAL;
  VALUE-LIST:(pilot,
      passenger, FTO,
      CBC, ASBG, ASIC,
      ASSV, ASCH )
END VALUE-TYPE valor-role;
```

Figure 3.3: Concepts using CML.

* Crew License: A web service containing the flight license of the crew assigned to the current flight. It is available to pilots only.
* Safety Demo: An informative document about safety available to pilots and passengers only.
* Weather Map: An informative document available to pilots, FTO and CBC.
* News: Access to the URL of major Spanish, English and French newspapers. It is available to pilots and passengers.
* Passenger Control Report: A web service that delivers the passenger list of the current flight. It is available to pilots and FTO.
* ID: A barcode used to improve the ID data capture. It is available to pilots, CBC and FTO. Code 39 is used to encode the one dimensional barcode

These concepts can be represented in different ways. Figure 3.3 shows the representation of the user concept using CML language, where the characteristics are represented by means of " attributes" unlike other approaches, where functional information is included as well as the characteristics.

Another way of representing these concepts is to use a domain ontology. The domain ontology is defined by means of three sets: a set of terms, a set that contains their definitions (typology) and a third set of relations between the terms (taxonomy). The ontology provides the explicit conceptualization of the domain terms to support the knowledge base implementation which is prepared for use by the applications to perform different tasks. Figure 3.4 represents the specific conceptualizations for the particular domain of an airport. CKADS has an extension of the initial phases for the development of ontologies that covers its entire life-cycle, from the feasibility study until the maintenance phase. Figure 3.4 contains the representation of the domain knowledge for the airport.

Once we the concepts have been defined, relationships can be established between them. We need to represent an antecedent and a consequent in order to define the different type of rules in our domain schema. It is also necessary to represent the connection symbols used

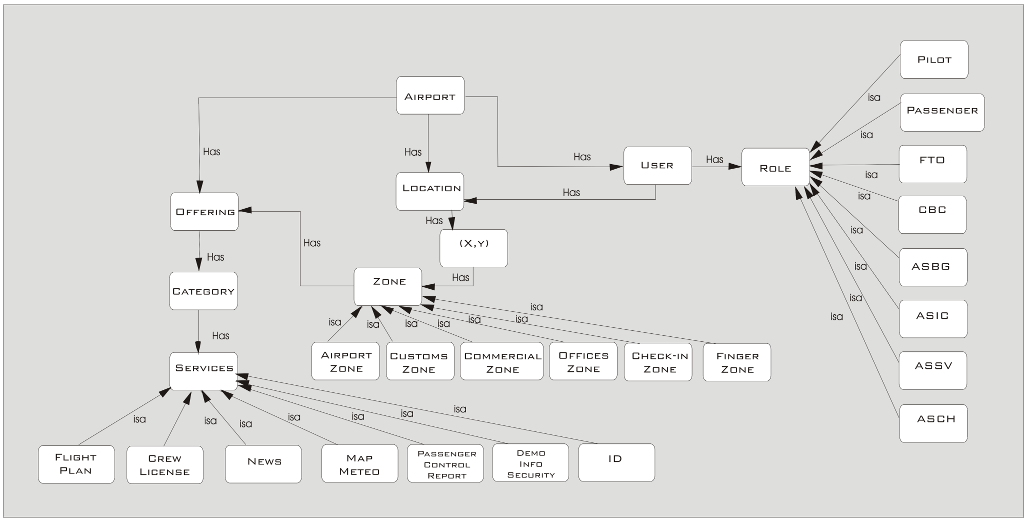


Figure 3.4: Domain knowledge in the Airport KBS.

to connect the antecedent to the consequent. The input/output curve represents a global relationship or a group of partial I/O relationships. The Domain Knowledge is gathered from these relations between input ranges and output ranges. For instance, if we know a passenger user of our domain is in a check-in zone, he will be offered the check-in service. In this case, the rule can be assessed as:

Rule inference: *If a user with a certain role is in a certain zone at a certain time, then he or she will be made an offering.*

Rule inference: *Given an offering and a user category, then some services are provided to the user and others are not.*

3.1.2.2 Inference layer

After defining the Domain Ontology, the domain dynamic component must be obtained. This dynamic aspect deals with the system input/output behaviour that is stated as a set of production rules. In this layer we will describe how the static structures defined above will be used to develop the reasoning process.

Inferences are completely described through a declarative specification of their inputs and outputs (dynamic roles). Returning to the CML language representation, Figure 3.5 shows the CML specification of an inference, where the input is a role "case" representing the knowledge elements of the domain and the output is the role "abstracted-case" representing the qualified description of the input. Figure 3.6 shows the "case" and "abstracted-case" roles representing the smoking-related characteristics in the airport domain.

```
INFERENCE abstract ;
  OPERATION-TYPE :ABSTRACT;
  ROLES:
    INPUT: case;
    OUTPUT: abstracted-case;
    STATIC: knowledge  -abstraction;
  SPECIFICATION:
  "Every time this inference is executed, it generates a transformation of
   the entrance data producing a more qualified description of the case"
END INFERENCE abstract;
```

Figure 3.5: CML specification of an inference.

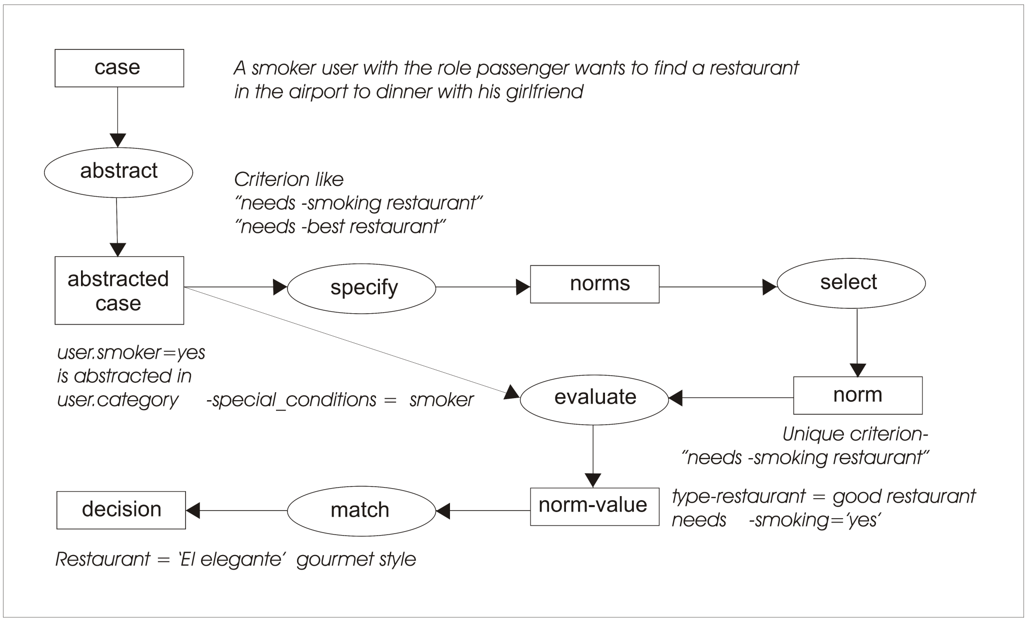


Figure 3.6: Inference diagram for decision making on choice of an airport restaurant

3.1.2.3 Task layer

When both the Domain Ontology and Dynamics have been defined, we have to divide inferences into tasks, where tasks are similar to traditional functions but the data manipulated by the task are described domain-independently. They describe the input/output and how a task is performed through an ordered decomposition into subfunctions, like another task, inference and transfer function. They form a small program or algorithm that captures the expert's reasoning strategy. The particularity of CKADS is the partial reuse of knowledge models in new applications, proposing a catalogue of task templates comprised by a tentative inference structure, a typical control structure and a typical task-based domain schema, which are specific for a task type. Of the task types suggested by CKADS, we consider our problem as a special case of an assessment task template, where an assessment problem consists of finding a decision category for a case based on domain-specific norms. The typical control

```
while new-solution abstract(case-description -> abstracted-
  case)
do
case-description := abstracted-case;
end

while specify(abstracted-case -> norms);
 repeat

 select(norms -> norm);
 evaluate(abstracted-case + norm -> norm-value);
 evaluation-results:=norm-value union evaluation-results;

 until has-solution match(evaluation-results -> decision);
```

Figure 3.7: Control structure.

structure suggested by CKADS for assessment problems is shown in Figure 3.7.

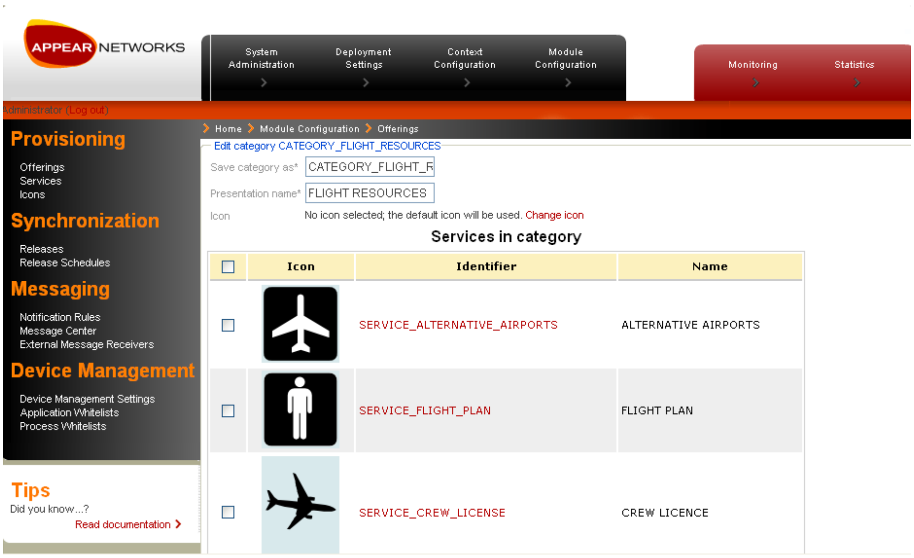
3.1.3 Appear context configuration using CKADs

At the beginning of the section we explained we will support the description of some sections using an airport domain that is detailed in Chapter 4. Here we explain how Appear represents the context and how the KBS for the airport domain was defined (Sánchez-Pi et al., 2007).

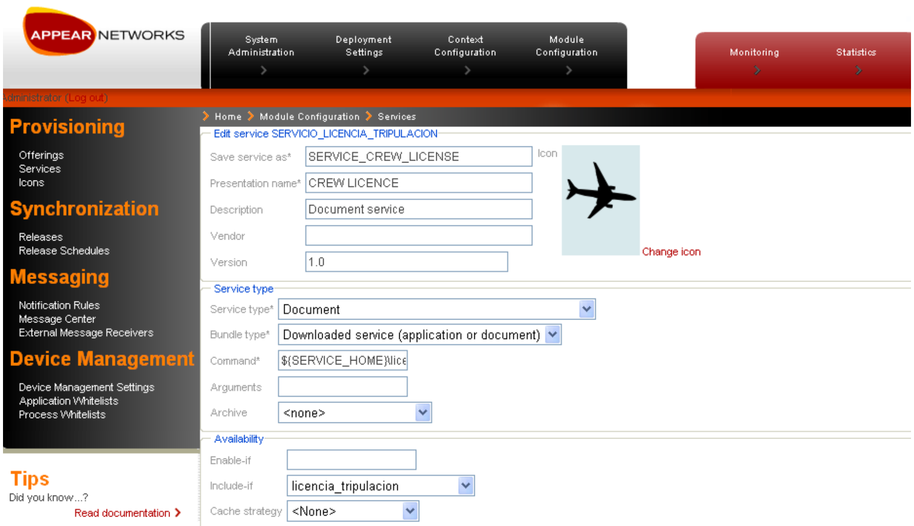
3.1.3.1 Domain knowledge configuration in Appear

The offering mapping is where offerings are mapped to zones and times. A mapping consists of a condition and an offering. The meaning of the mapping is that when the context of the user's device matches a condition, the client will be eligible for the offering During the context configuration procedure, we need to define every concept in our domain knowledge and represented in the ontology described in Figure 3.4. These concepts are:

- Service: An application or data available to a client
- Category: A set of logically grouped services
- Offering: A set of available applications for a client
- Zone Mapping: Where offerings are mapped to zones and times
- Roles: Used to grant users access to specific categories. It is possible to specify one or more groups that have access to a category.
- Users: Users are only used if the system requires login.



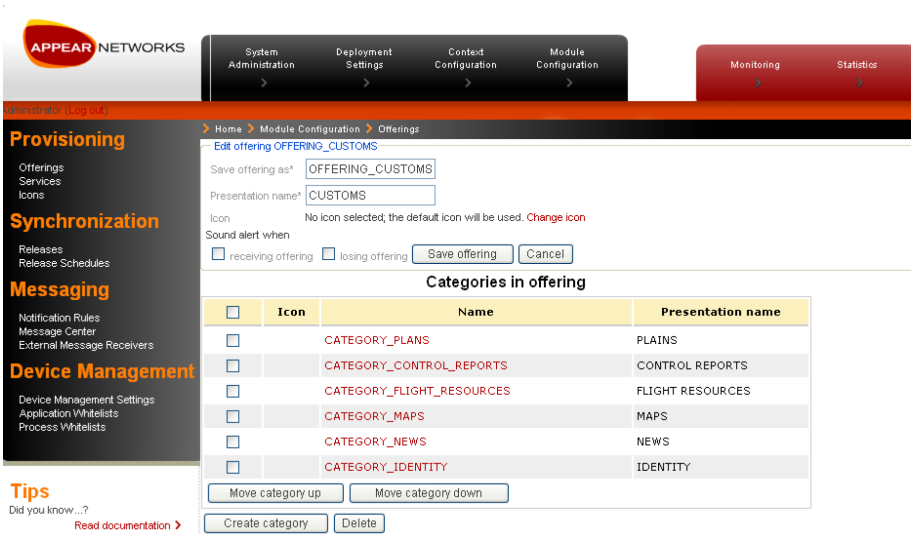
(a) Definition of a category with Appear.



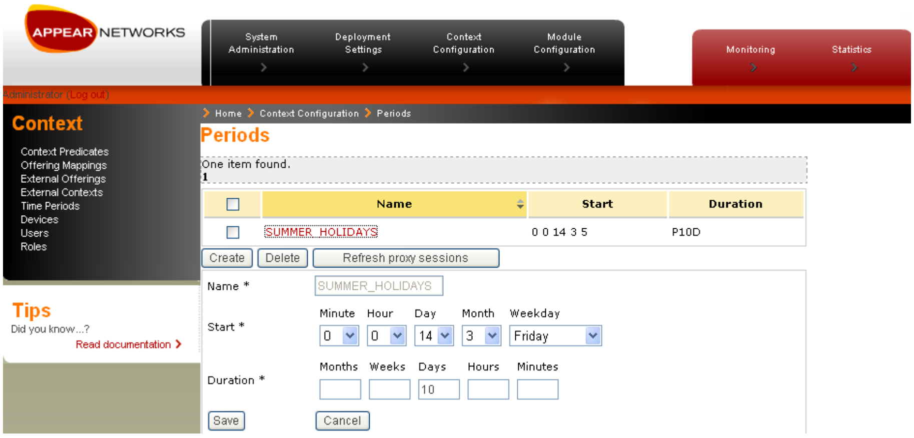
(b) Definition of a service with Appear.

Figure 3.8: Definition of different elements in Appear (contd. in Figure 3.9).

There are several setting values for the context elements that form the condition. They are: Zone name, Time period, IP pattern and Roles. The Zone name is known in the positioning system. Once the positioning system gives the location coordinates of the user's device, the plug-in defined in Appear, which in our domain is divided into zones, translates and assigns a Zone name to these coordinates. The Time period assures that the mapping is valid only when the time period is valid. The IP pattern restricts access to users whose IP addresses comply with a particular pattern. The name of the offering that should be used when a condition matches is typed in the Offering field. Icons, offerings, categories and services are the other concepts that need to be defined in order to complete the offering mapping. Offerings, categories and services can all use icons that will be displayed in the



(a) Definition of an offering with Appear.



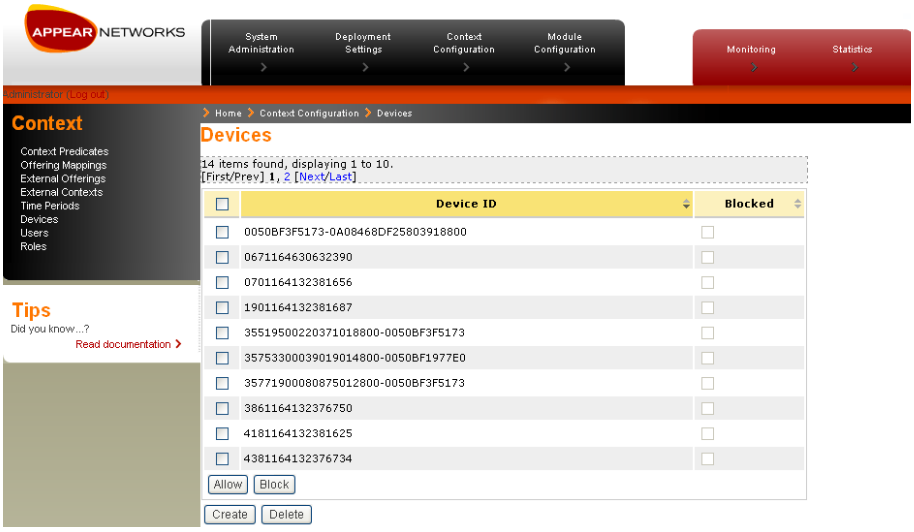
(b) Periods definition with Appear.

Figure 3.9: Definition of different elements in Appear (contd. in Figure 3.10).

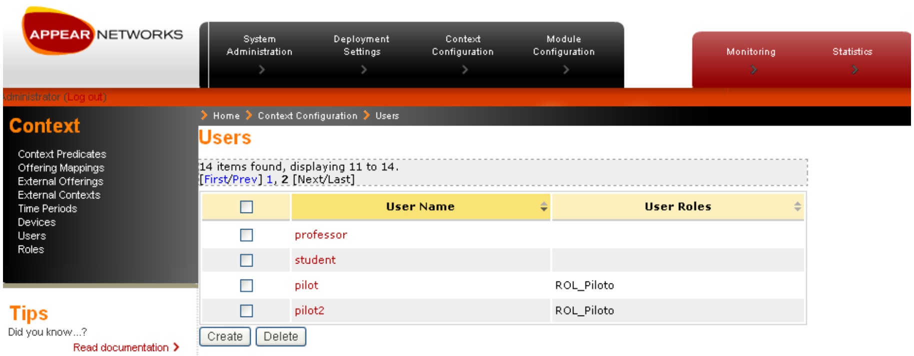
user's interface. Icons are grouped by their usage into icons for services, icons for categories and icons for offerings. A category is a group of services. A category restricts access to services, see Figure 3.8a. Services in Appear are easily configured, see Figure 3.8b, and it is possible to create services of different types. There are templates for creating document services, web services, native services (Windows services) and Java services.

For a document service the file must be uploaded; for a Web Service, the URL for the web page must be specified; for native services, the executable file must be specified, and for Java services, the jar file and the main class should be specified. And, finally, the offering concept groups categories and services. One offering can contain several categories 3.9a, and each category can contain a number of services. The user's device can only display one offering at a time.

A time period, as its name suggests, consists of a name and a definition of a period of time. Figure 3.9b shows the configuration of a period of time. Period name, start time and



(a) Devices definition with Appear. Enable or block access.



(b) Users definition with Appear.

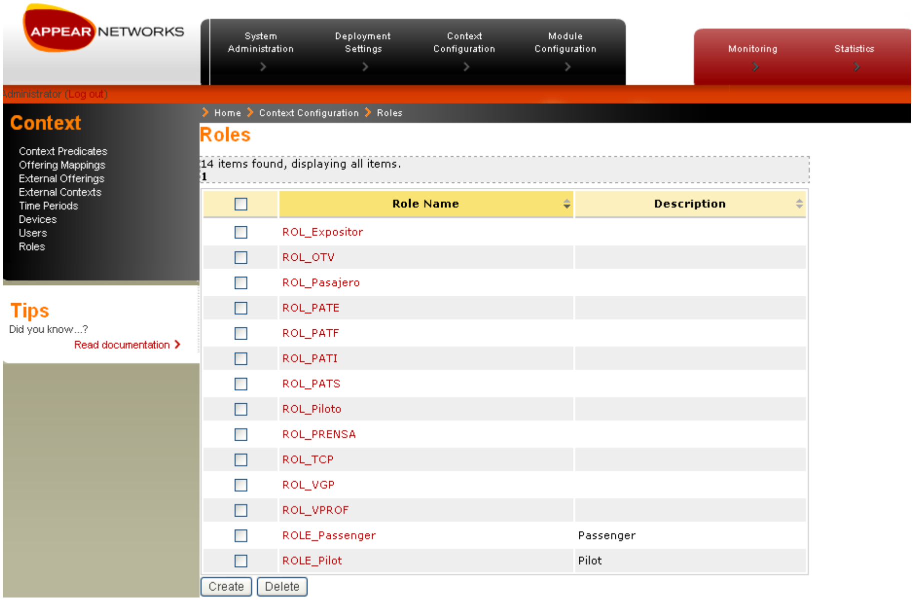
Figure 3.10: Definition of different elements in Appear (contd. in Figure 3.11).

duration are concepts that need to be configured.

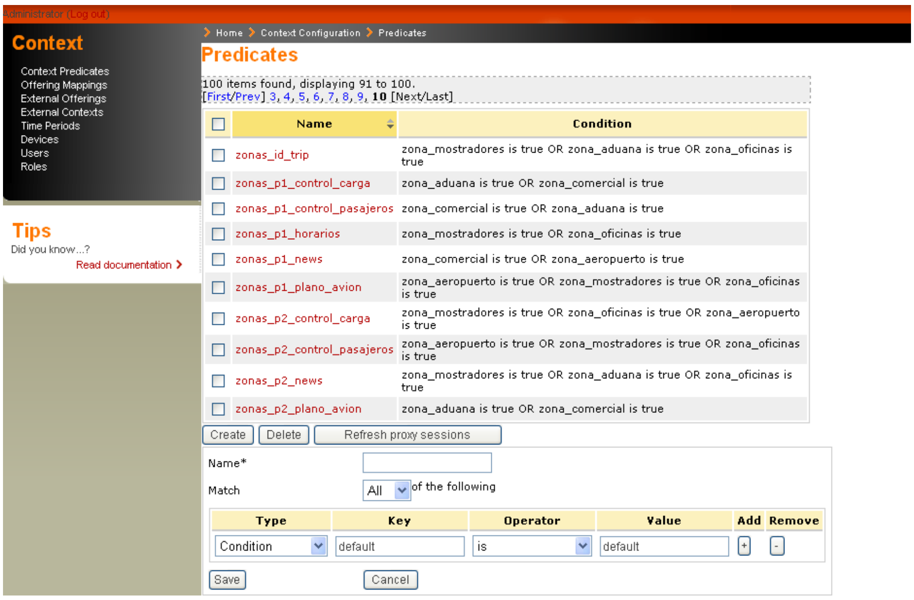
The first time a device connects to the system, it is assigned a unique key by the Appear Context Engine. After that, each device can be enabled or disabled individually, as shown in Figure 3.10a.

The services are automatically provisioned in the Appear Context Engine as each user's context is identified: role, zone, location, time period, etc.. After defining the users, Figure 3.10b, we need to create the roles, Figure 3.11a that are used to give users access to specific categories of services.

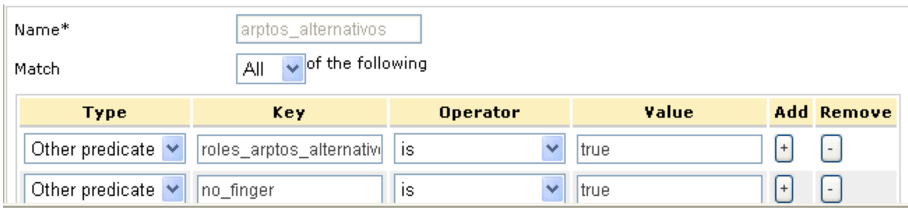
It is possible to specify role restrictions on categories. If a category has no roles, every user has access to the category. To restrict access, it is possible to specify one or more groups that have access. This means that only users that belong to the specified role can gain access to the category. Other users will not see these categories in the client.



(a) Roles definition with Appear.



(b) Predicate definition with Appear.



(c) A predicate in Appear.

Figure 3.11: Definition of different elements in Appear.

3.1.3.2 Inference knowledge configuration in Appear

As explained abstractly, there are concepts in Appear, like predicates, context predicates and conditions, figure 3.11b, that are combined to create rules which are part of the rules engine in the Appear Context Engine. This is what determines which service is available to whom, and where and when it should be pushed to the user's device.

A new predicate is created as shown in Figure 3.11c, adding a condition with a keyword in the key field, selecting the required condition ("is", "is not", "contains") and then typing an appropriate value to test against the value field.

Relations between corresponding input and output segments could be assessed as:

Rule Inference: *Given a zone and time of the client, then the client uses some offering.*

There are several setting values for the attributes of the domain concepts that form the condition. They are: Zone name, Time period, IP pattern and Roles.

- The Zone name is known in the positioning system. Once the positioning system gives the location coordinates of the user's device, the plug-in defined in Appear, which in our domain is divided into zones, translates and assigns a Zone name to these coordinates..
- The Time period assures that the mapping is valid only when the time period is valid.
- The IP pattern restricts access to users whose IP addresses comply with a certain pattern. The name of the offering that should be used when a condition matches is typed in the Offering field.

Icons, offerings, categories and services are the other concepts that are need to be instantiated in order to complete the offering mapping, since an offering consists of several services, but a category restricts access to some of these services corresponding to the offering. For this reason, they form another type of inferences.

Rule Inference: *Given an offering and client category, then some services are provided to the user and others are not.*

Context information in the system is used throughout the entire life-cycle of the service. The rules engine filters and determines which is the appropriate service to be pushed to each user, in the right time and at the right place.

In our architecture we have included a plugging in order to link dialogue system modules. In our architecture, we have incorporated a Context Manager. This module deals with creating, updating and loading the context information by means of specific profiles associated to each user. It also communicates this information to the DS modules during the interaction. The information of a user profile can be classified into three different groups:

- General user information. This set includes personal information, such as user's name, gender, age, current language, skill level when interacting with dialogue systems, pathologies or speech disorders, if any, and so on.
- General statistics. This second group comprises the number of dialogues and dialogue turns, their durations, the date of the last interaction with the system, etc..

- Usage statistics and user privileges. This set stores the counts of each action over the system that a user performs, and a mark of user clearance for each possible action. The set is split into different statistic groups, which are defined according to the specific domain-knowledge we have. This new subdivision allows the system to infer the preferences of each user among the different statistic groups, and among the different items belonging to the same group.

Static context is represented and managed in the AIQ layer using concepts like role, zone, and location. This context information is mapped into zones and times to get the right services. A mapping consists of a condition and an offering. There are several setting values for the context elements that form the condition: Zone name, Time period, IP pattern, and Roles. The Zone name is known in the positioning system. Once the positioning system gives the location coordinates of the user's device, the plugging defined in Appear translates these coordinates and assigns a Zone name to them. The Time period makes the mapping valid only when the time period is valid. The IP pattern restricts the access to users whose IP addresses fulfil a certain pattern. The name of the offering that should be used when a condition matches is typed in the Offering field. The rest of features defined for representing the user profile are managed by the DS using the user profiles. The transmission of the context between the different modules is carried out by sending XML packages based on the OASIS Web Services Context Specification 4.

Until here we have provided with a description of knowledge modelling using CKADS and the implementation of the centralized system using Appear but for large developments there is a need to provide with a distributed solution in order to increase feasibility, scalability and fault tolerance. So in next section we will introduce a distributed implementation based on intelligent agents in order to provide the context aware services.

3.2 Distributed system based on intelligent agents

Transparency, autonomy and proactivity are characteristics valuables for AmI applications. People need to be like in social context while developing ordinary activities As O'Hare claimed,(O'Hare et al., 2004) intelligent agents seem to be the appropriate solution for AmI environments because of their own properties. If we assume that agents are abstractions for the interaction within an AmI environment, one aspect that we need to ensure is that their behaviour is coordinated. So, the use of agents are a key enabler in the delivery of AmI applications. That is the main motivation and the one explained at the end of last section, which lead us to introduce and implement intelligent agents techniques for the provisioning of context aware systems.

In this section, in order to model and represent knowledge of an AmI environment, we present an ontology-based approach and also the Gaia methodology to analyse and design our agent based system.

3.2.1 Ontology approach

In order to represent knowledge and in order to provide the system with the reasoning capacity and as explained in previous section, we used an ontology approach. For building the ontology, we have followed Noy and McGuiness proposal which consists on an iterative process based on the methodology proposed by Gruninger and Fox (Gruninger & Fox, 1995) who defined the competency questions used in the scope and goal step, and the development of the classes hierarchy based on Top-Down and Bottom-Up strategies. The applied steps for developing the proposed ontology are the following (Noy et al., 2001):

- Determine ontology goal: it is important to have clear requirements and the intention of the ontology use.
- Consider the integration of existing ontologies: reusing ontologies is a requirement in order to interact with other applications. In the case that no relevant ontology can be reused, the better option is to develop a new ontology from scratch by taking in account all categories of context.
- Ontological acquisition: defining the ontology implies a process of ontological acquisition, which consists on identification the key concepts and relationships of interest domain. A Top-Down strategy, a Bottom-Up strategy, or a combination of both of them, can be used for the ontological acquisition step. Top-Down starts with a general vision of the problem, and go down to instantiate the specific concepts of the domain and Bottom-Up strategy obtains a high level abstract conceptualization from different specific domain applications. The combination is needed to check the matching between the high level of conceptualization and the lower level.
- Codification process: in this step the ontology is created using Protégé tool (Project, 2000).

3.2.1.1 Ontology high level definition

Normally, ontology represents a conceptualization of particular domains. In our previous works (Fuentes et al., 2006a,b), we studied the problem of context definition in ubiquitous applications. The proposal of our researches is a high level conceptualization ontology definition to cover whole context in several heterogeneous domains. The intention of this particular contribution is to apply a high level definition to solve the airport problem. Furthermore we evaluate the effectiveness of the previous researches by checking the matching between this high level for dynamic environments, and the following lower levels presented in this paper for the airport domain.

The high level ontology definition that we have described follows the categorization defined by Schilit (Schilit et al., 1994), that divided contextual information in computing context (network, devices, etc.), user context (preferences, location etc.) and physical context (temperature, traffic, etc.). The ontology definition gathers these concepts and their properties and relationships for accomplished this contextual definition.

Following the previous defined steps for building an ontology, our proposal meta-ontology focuses mainly on all concepts definition in order to be valid for any environment or domain. These ontological high level concepts can be described as follows:

- Framework is the general application concept which includes high level system concepts and defines what is the current environment or domain of the system. It has two slots:

Sector and Event, that represents, system sector (mobile, technology, commercial etc.) and the current event (fairground, airport etc.), respectively.

- Location represents the (x, y) coordinates of any place, participant or object.

- Spatial region define the environment area. This concept represents the map or NxM area, and it is composed by buildings, floors and segments with a range of positions each one. For example, segment1 is a segment with the range of positions: (1, 2) (1, 3) in the second floor of the Building A.

- Temporal Region concept gathers time information about users in any location in spatial region. Temporal region represents user date (dd/mm/yyyy) and hour (hh:mm) when he is in a specific position inside the map.

- Place concept represents interest points in the environment. Places can be specific zones, stands, expositors etc., for instance.

- Participant concept refers to people or companies that play a role and has a profile in the system. Participants can be people and companies.

- Service concept can be any kind of system provision offered to users referred to contextual information. A service could be a notification in user device about preferential user product. The profile is a Participant's subclass, and it gathers preferential product, firm, price, model etc.

- Product stands for any kinds of information or object that users requires to be informed.

- Device concept gathers information about different user's devices in which the system works. An example of used device is a PDA or a smart phone.

- Predicate stands for facts or expressions that say something about the world state: located, belongs-to etc.

- Agent Actions represents special concepts that indicate some actions developed by agents, like communicative acts or ACL messages: sell, communicate, recommend, for instance.

- Agent Identification identifies entities or agents.

3.2.1.2 Environment definition

Here, we will make reference to the airport domain which will be completely detailed in next section. The ontology definition for the airport problem is based on this high level of conceptualization created to cover several environments. Furthermore, solving the airport ontology definition let us analyse the accuracy of the meta-ontology, in order to check the matching between the lower levels of conceptualization the high level of meta-concepts. According to the problem definition, the lower levels of the ontology can described as follows:

- Place: this concept refers to any place that can be defined in the airport environment. In this case the main place is the Airport concept, and inside of it: Check-in-Desk, Finger, Offices, Commercial and Customs as Figure 3.12a. Moreover, it is possible to define more places according to others needs added to the problem: Souvenir-Shop, Smoking-Point, for instance.

- Participant class is organized in two main subclasses: People and Company. At the same time, Role and Profile are People subclasses. Role subclass match with the Role Definition of the problem since it is possible to define the roles: Role-FTO, Role-CBC, Role-ASBG, Role-Pilot, Role-Passenger, Role-ASIC, Role-ASSV, Role-ASCH as shown Figure 3.12b. These roles are associated to people concept, as well as profile concept that includes personal characteristics. People concept is related with Location concept, since every person is located somewhere in the environment.

- Services concept includes several subclasses corresponding to defined categories in the airport problem: Map-Type, Control-Report, Flight-Resources, ID, News and Plans. All services depending on its respective category (Map-Meteo, French-News, etc.) is defined as Products that these services can provide. So, Service hierarchy in the ontology represents only the service categories in airport definition, Figure 3.12c.

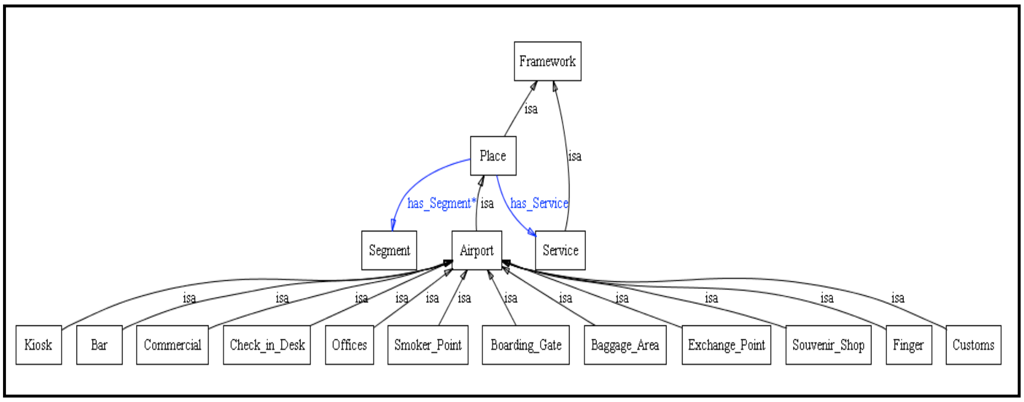
- Predicates in the ontology are categorized in two subclasses according to the airport problem: Role-Predicate and Zone-Predicate. The hierarchy of the Role-Predicate is shown in Figure 3.13a. The respective representation of the Zone-Predicate is defined in the ontology.

- Products are all concepts that a service can provide. In this case, products are organized following the service categorization defined in the airport problem. The service definition is translated as products, offered to clients, in the ontology description. It includes several subclasses attending to these categories: Map-Type, Control-Report, Flight-Resources, ID, News and Plans. It is possible to add other products as Flight-Information depending on the necessities of the problem. Figure 3.14

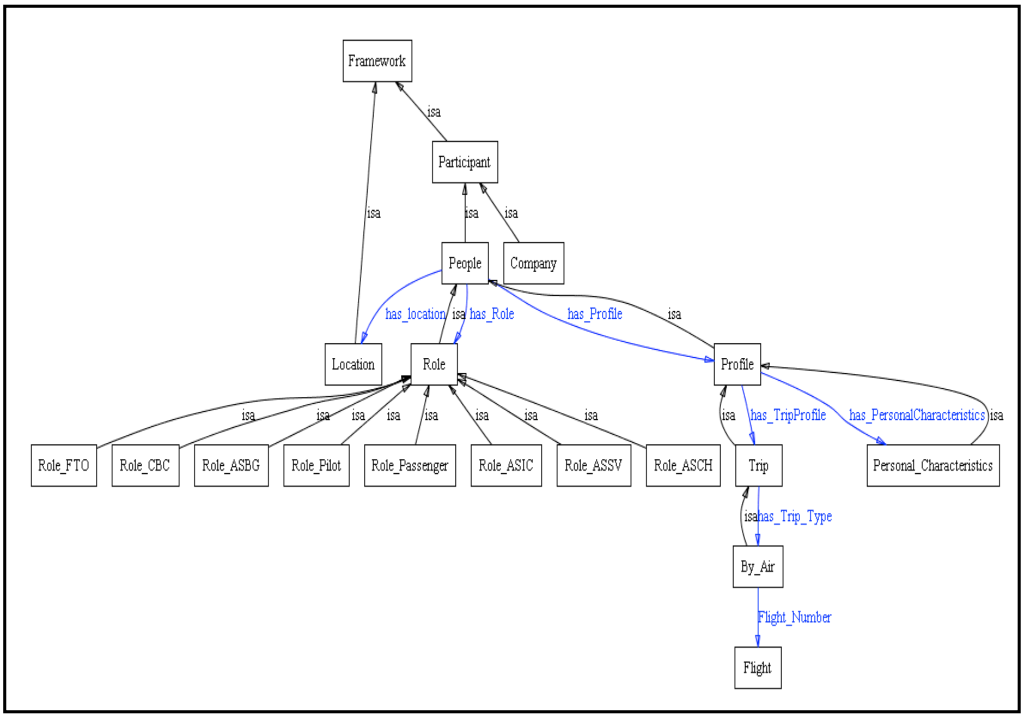
For the effectiveness of the ontology definition, it is necessary to define the relationship between People Roles, Location, Places, Spatial Region, Services and Products. A client (People) with the passenger role (Role is a subclass of People) is located in some position in the Spatial Region, represented by segments in the building floor, which corresponds to one or more Locations. This passenger is moving through the environment, and when he is near to a place, he receives in his device a service offered some product. So, People concept is related with Location concept by the "has-Location" slot, the Segment subclass in Spatial Region is related to Location concept too, so, indirectly People and Segment are connected. Moreover, Places includes a "has-Segment" slot that connects the Place with the Segment. Every place offers a service and every service provide products, so Place, Service and Product concepts are related too. Figure 3.13b shows some of the main relationships previously described.

3.2.2 Multiagent approach to AmI

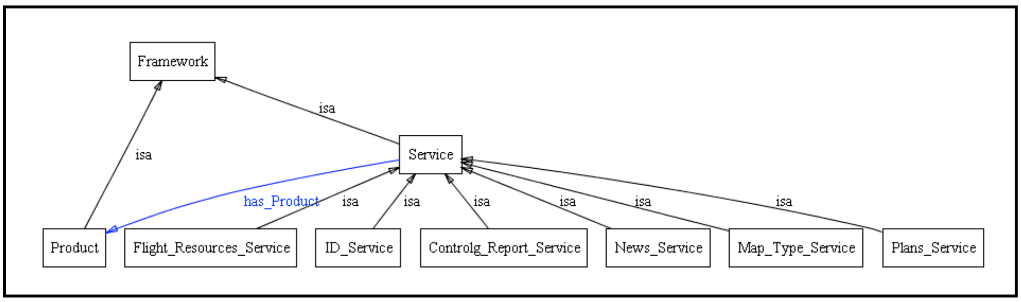
The ubiquitous environments ideally fit with the agent paradigm the same way ambient intelligent paradigm does. Multi-agent systems support complex interactions between entities,



(a) Place hierarchy in the ontology.

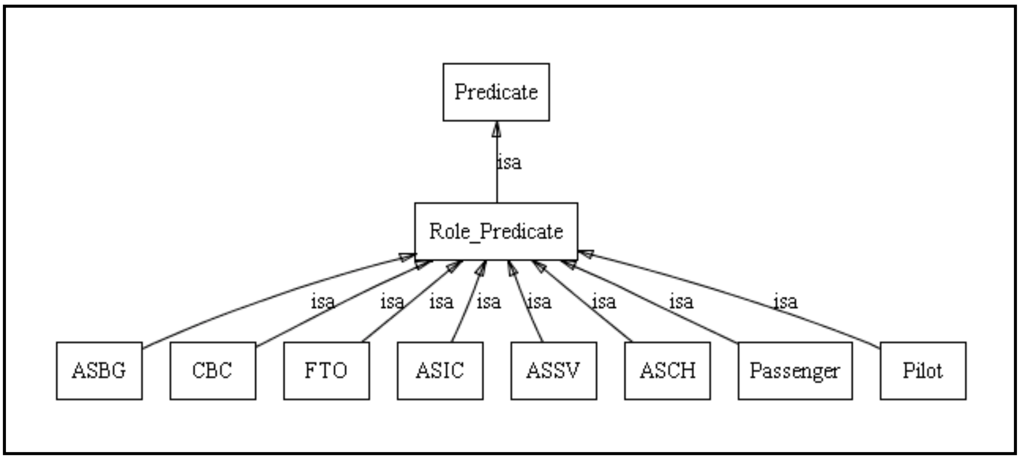


(b) Participant hierarchy in the ontology.

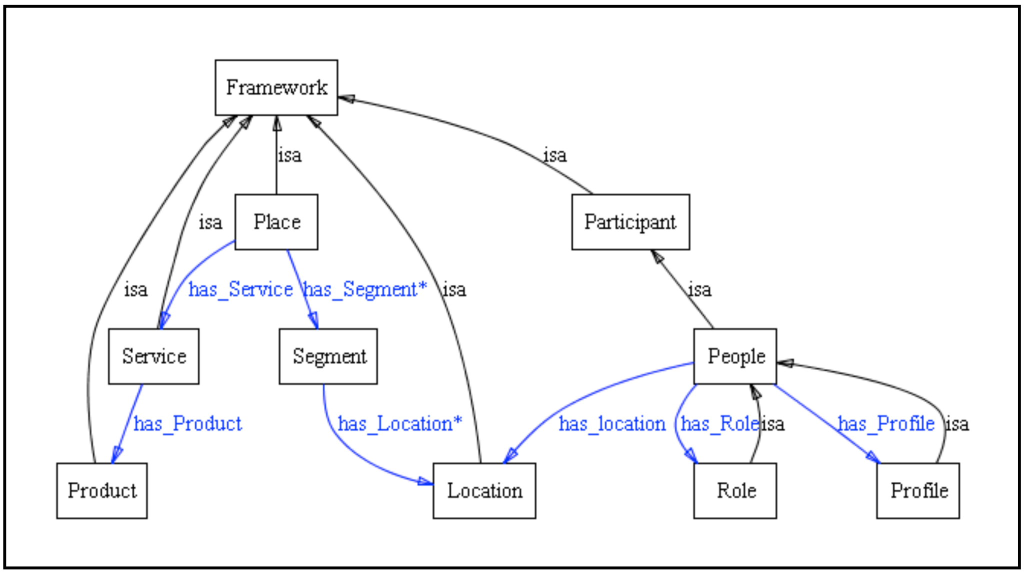


(c) Service hierarchy in the ontology.

Figure 3.12: Definition of different elements in the ontology.



(a) Role predicate hierarchy in the ontology.



(b) Relationship between ontology concepts.

Figure 3.13: Definition of different elements in the ontology (continued).

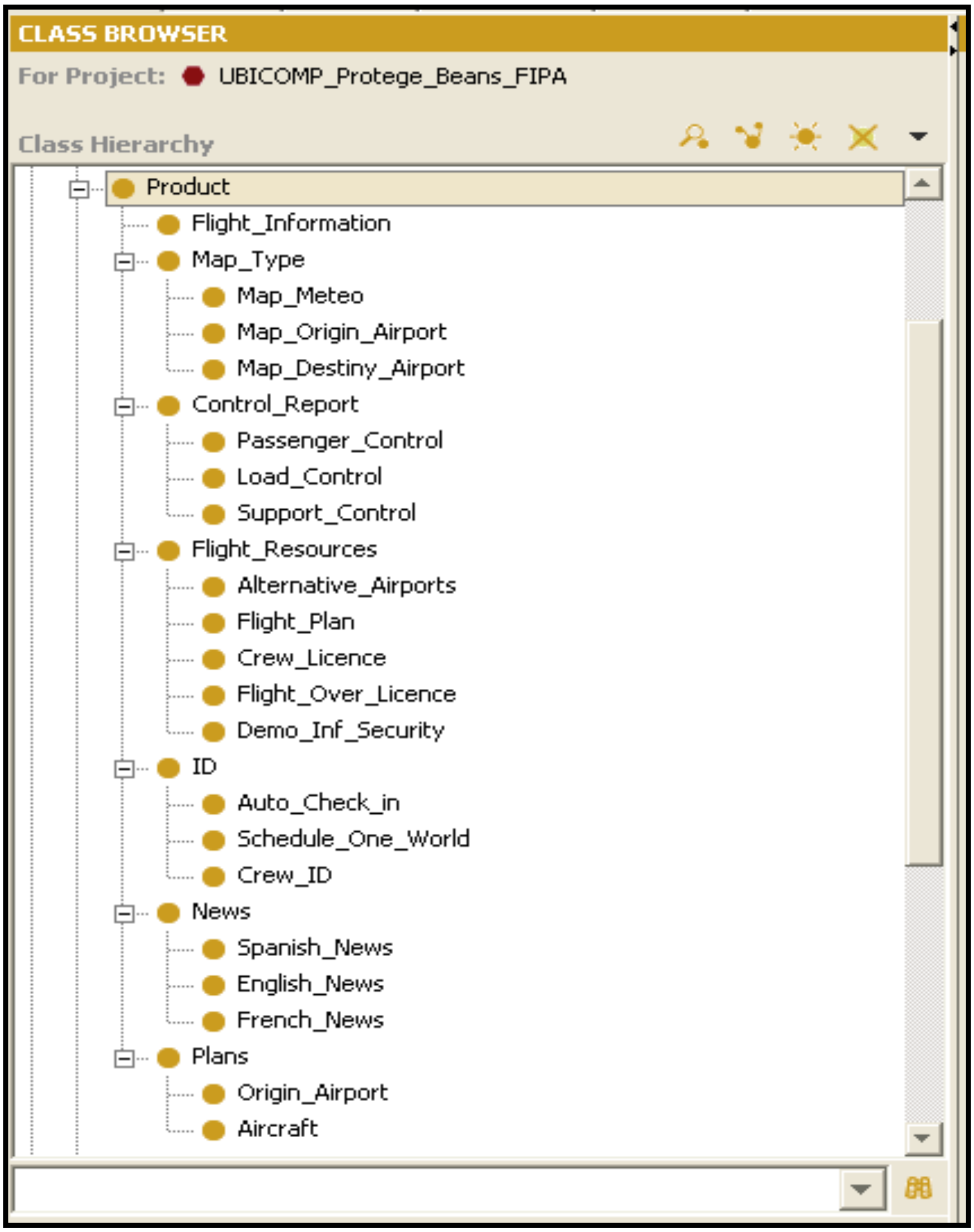


Figure 3.14: Product hierarchy in the ontology.

using high level semantic languages. Such a feature is essential in Ambient Intelligence environments dealing with heterogeneous information from physical sensors and users preferences. Integration of such data is only possible at a higher level where all kind of information (about context) is expressed semantically.

Ambient Intelligence (AmI) emphasizes on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. For this reason, AmI systems usually consist of a set of interconnected computing and sensing devices which surround the user pervasively in his environment and are invisible to him, providing a service that is dynamically adapted to the interaction context, so that users can naturally interact with the system and thus perceive it as intelligent (Kovács & Kopácsi, 2006).

To achieve this goal it is necessary to provide an effective, easy, save and transparent interaction between the user and the system. To do so, in the last years there has been an increasing interest in simulating human-to-human communication, employing the so-called conversational agents (López-Cózar & Araki, 2005). Conversational agents, which enhance agents with computational linguistics, have became a strong alternative to provide computers with intelligent communicative capabilities. There is a high variety of applications in which conversational agents can be used, one of the most wide-spread of which is information retrieval. Some sample applications are tourist and travel information (Glass et al., 1995a), weather forecast over the phone (Zue et al., 2000a), speech controlled telephone banking systems (Melin et al., 2001a), conference help (Bohus et al., 2007a), etc. They have also been used for education and training, particularly in improving phonetic and linguistic skills (Litman & Silliman, 2004).

3.2.2.1 MAS architecture

The proposed agent-based architecture manages context information to provide personalized services by means of users interactions with conversational agents. As it can be observed in Figure 3.15, it consists of five different types of agents that cooperate to provide an adapted service. *User agents* are configured into mobile devices or PDAs. *Provider Agents* supply the different services in the system and are bound to *Conversational Agents* that provide the specific services. A *Facilitator Agent* links the different positions to the providers and services defined in the system. A *Positioning Agent* communicates with the Aruba positioning system (Sánchez-Pi et al., 2007) to extract and transmit positioning information to other agents in the system. Finally, a *Log Analyser Agent* generates user profiles that are used by Conversational Agents to adapt their behaviour taking into account the preferences detected in the users' previous dialogues.

Eight concepts have been defined for the ontology of the system. The definition is: *Location* (XCoordinate int, YCoordinate int), *Place* (Building int, Floor int), Service (Name String), *Product* (Name String, Characteristics: List of Feature), *Feature* (Name String, Value String), *Context* (Name String, Characteristics: List of Features), *Profile* (Name: String, Characteristics: List of Features), DialogLog (Log: List of Strings). Our ontology also include six predicates with the following arguments:

Our ontology also includes six predicates with the following arguments: *HasLocation* (place, Position, AID), *HasServices* (Place, Position, List of Services), *isProvider* (Place,

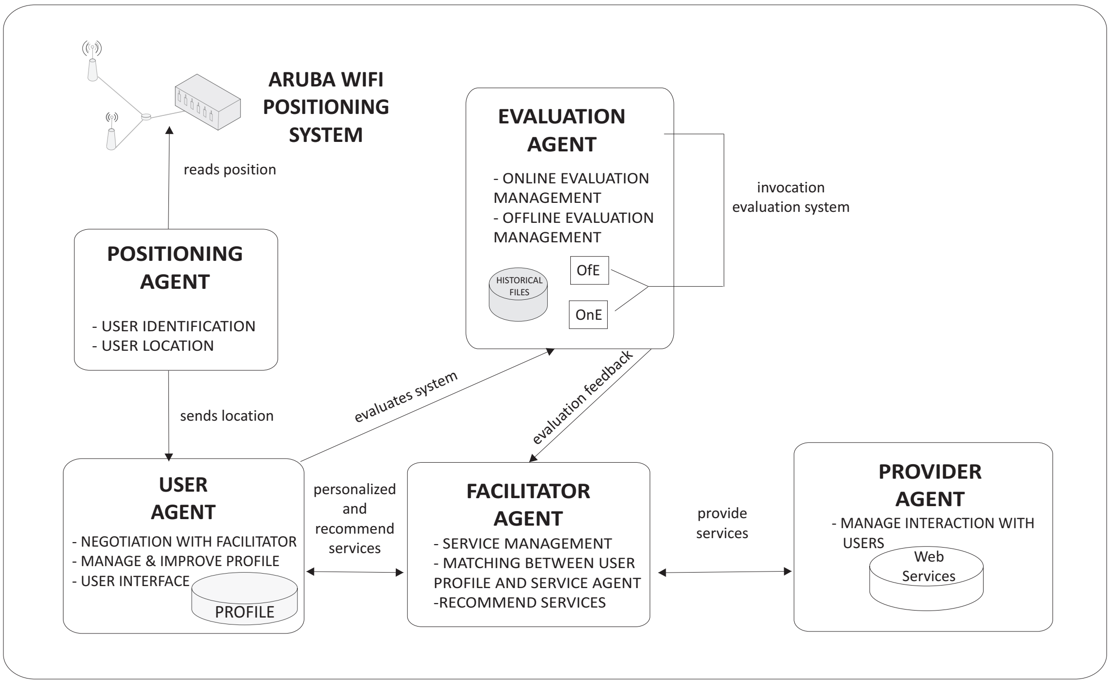


Figure 3.15: Schema of the multi-agent architecture.

Position, AID, Service), *HasContext* (What, Who), *HasDialog* (DialogLog, AID), HasProfile (Profile, AID), and *Provide* (Product, AID).

The interaction with the different agents follows a process which comprises the following phases:

1. The ARUBA positioning system is used to extract information about the positions of the different agents in the system. This way, it is possible to know the positions of the different User Agents and thus extract information about the different Providers Agents that are available for this location (see Figure 3.16).

2. The Positioning Agent reads the information about position (coordinates x and y) and place (building and floor) provided by the Aruba Positioning Agent (behaviour PositioningAgent.ReadFromAruba) by reading it from a file (behaviour PositioningAgent.ReadFromFile), or by processing manually introduced data (PositioningGui Interface) (see Figure 3.17).

3. The Positioning Agent (Positioning Agent.SendLocation) communicates the position and place information to the User Agent (behaviour UserAgent. ReceiveLocation) (see Figure 3.18).

4. Once a User Agent is aware of its own location, it communicates this information to the Facilitator Agent in order to find out the different services available in that location (behaviours UserAgent.FindServices and FacilitatorAgent.FoundServices) (see Figure 3.19).

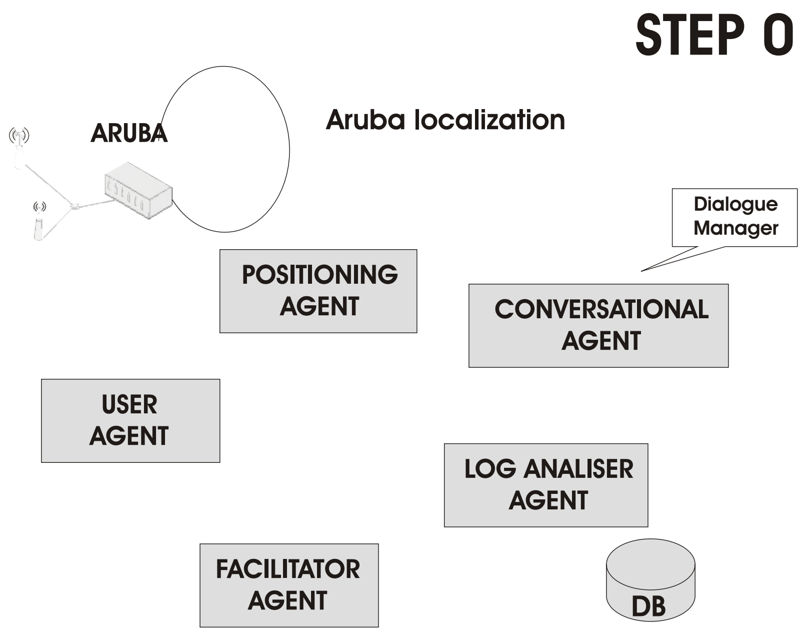


Figure 3.16: Step 0 — Agents interaction.

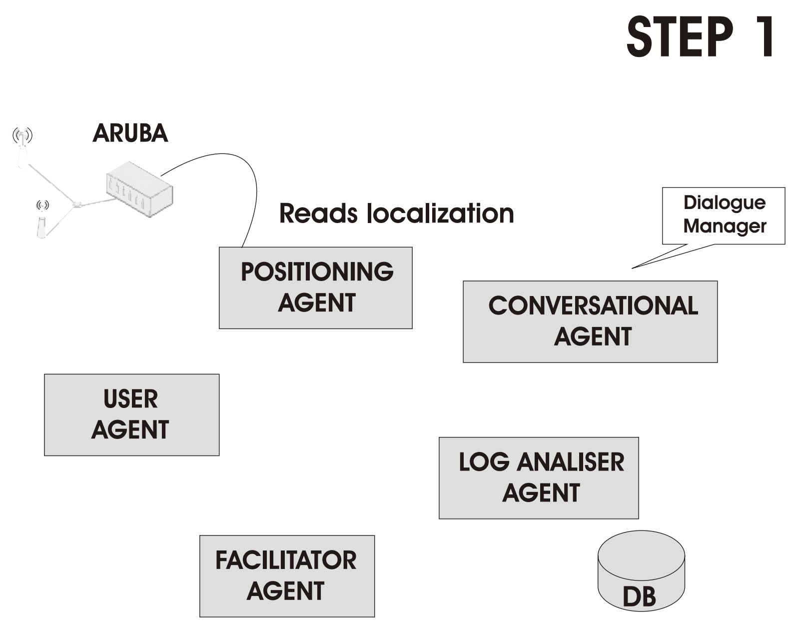


Figure 3.17: Step 1 — Agents interaction.

5. The Facilitator Agent informs the User Agent about the services available in this position (behaviours FacilitatorAgent.SendServices and UserAgent.ReceiveServices) (see Figure 3.20).

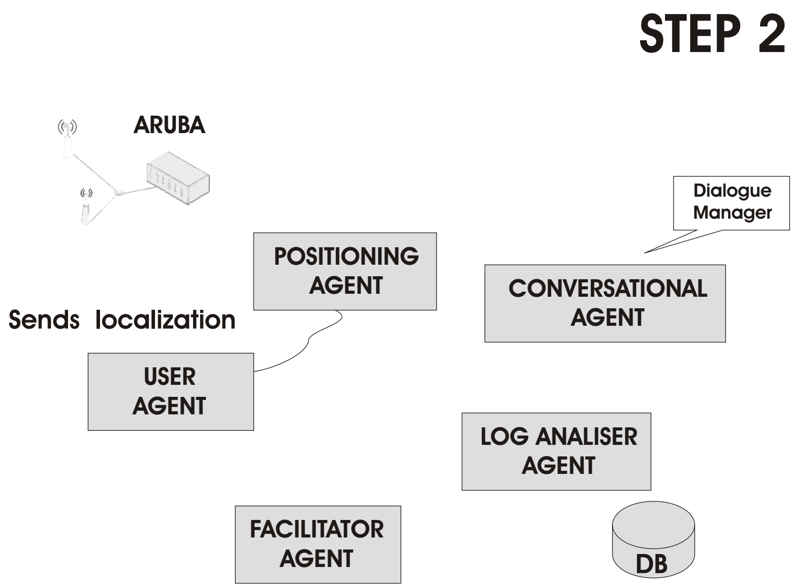


Figure 3.18: Step 2 — Agents interaction.

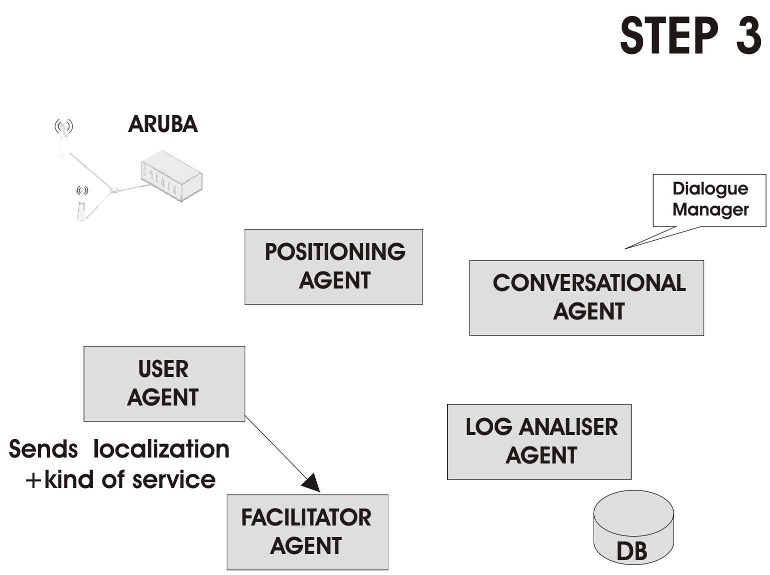


Figure 3.19: Step 3 — Agents interaction.

6. The User Agent decides the services in which it is interested (see Figure 3.21).

7. Once the User Agent has selected a specific service, it communicates its decision to the Facilitator Agent and queries it about the service providers that are available (behaviour UserAgent.FindProvider and FacilitatorAgent.FoundProvider) (see Figure 3.22).

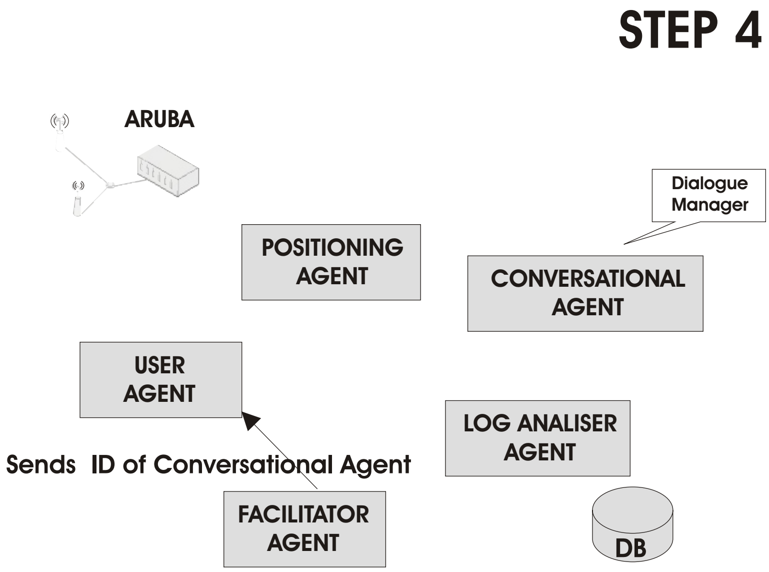


Figure 3.20: Step 4 — Agents interaction.

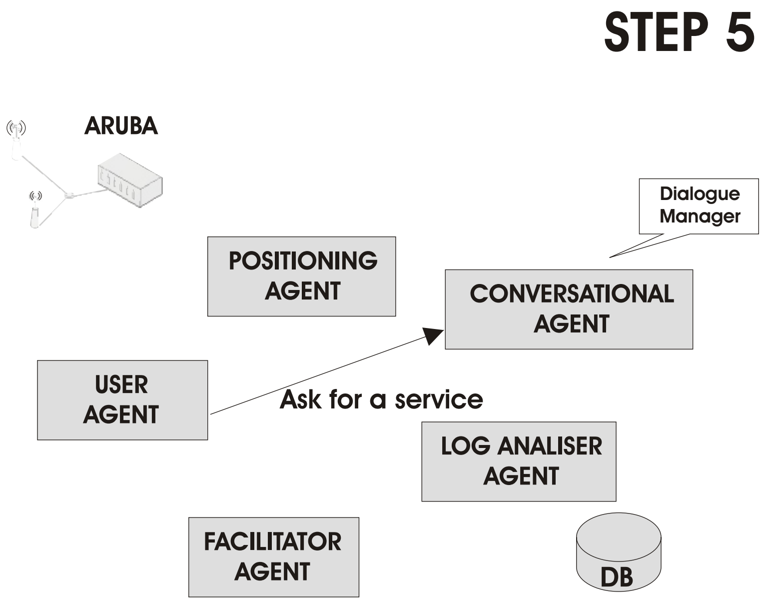


Figure 3.21: Step 5 — Agents interaction.

8. The Facilitator Agent informs the User Agent about the identifier of the Provider Agent that supplies the required service in the current location (FacilitatorAgent.SendProvider and UserAgent.ReceiveProvider) (see Figure 3.23).

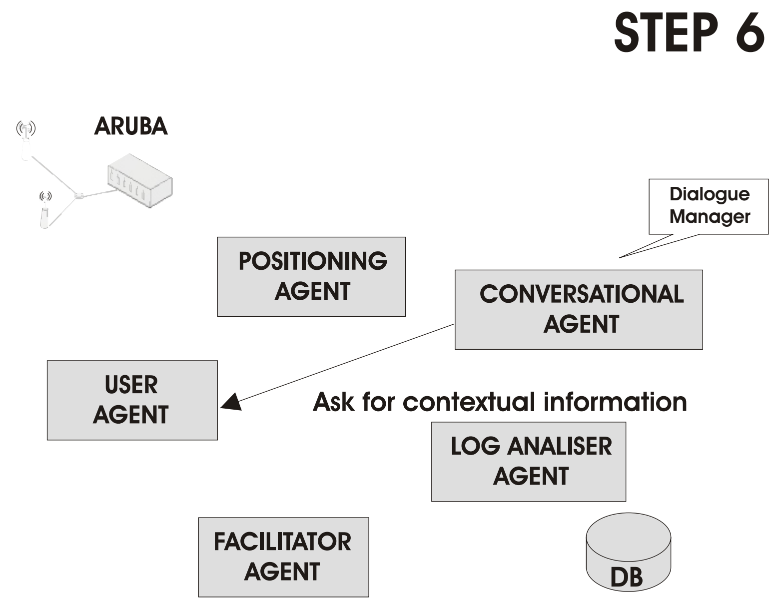


Figure 3.22: Step 6 — Agents interaction.

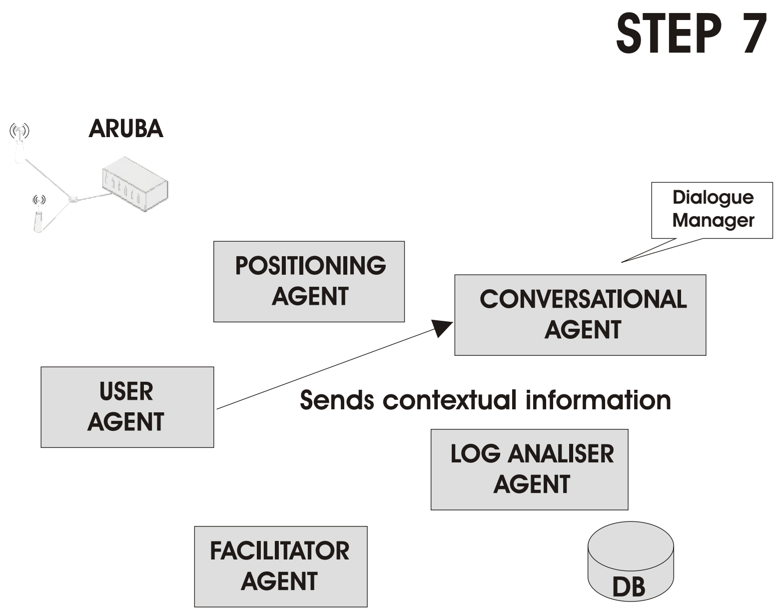


Figure 3.23: Step 7 — Agents interaction.

9. The User Agent asks the Provider Agent for the required service (behaviours UserAgent.RequestProduct and ProviderAgent.RequestedProduct) (see Figure 3.24).

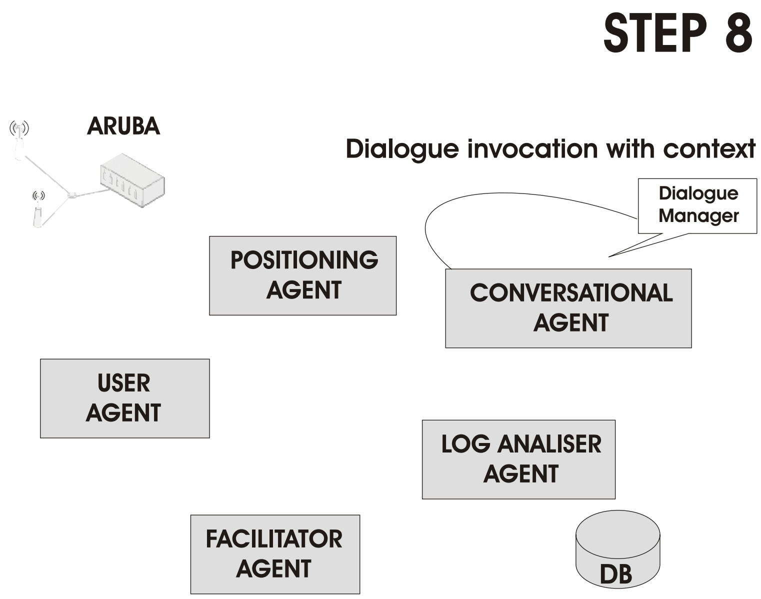


Figure 3.24: Step 8 — Agents interaction.

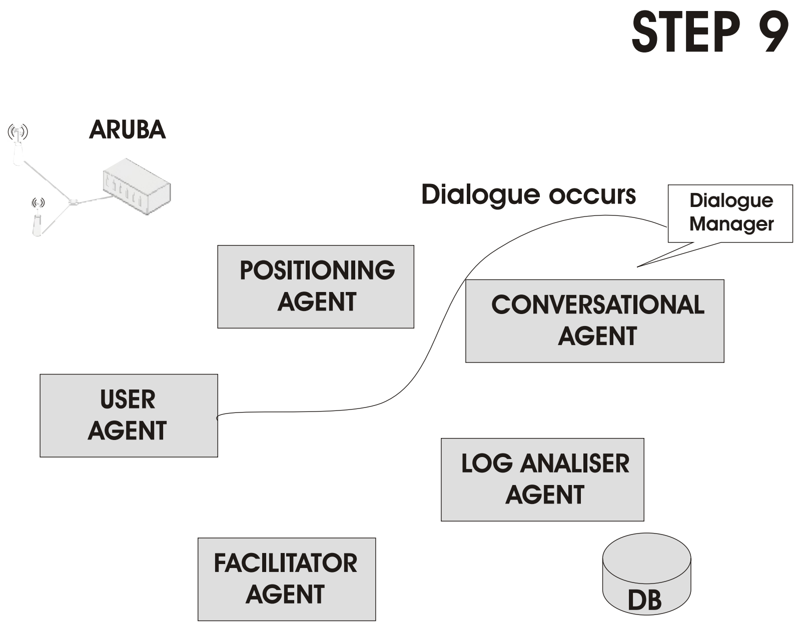


Figure 3.25: Step 9 — Agents interaction.

10. Given that the different services are provided by context-aware Conversational Agents, the Provider Agent asks the User Agent about the context information that would be

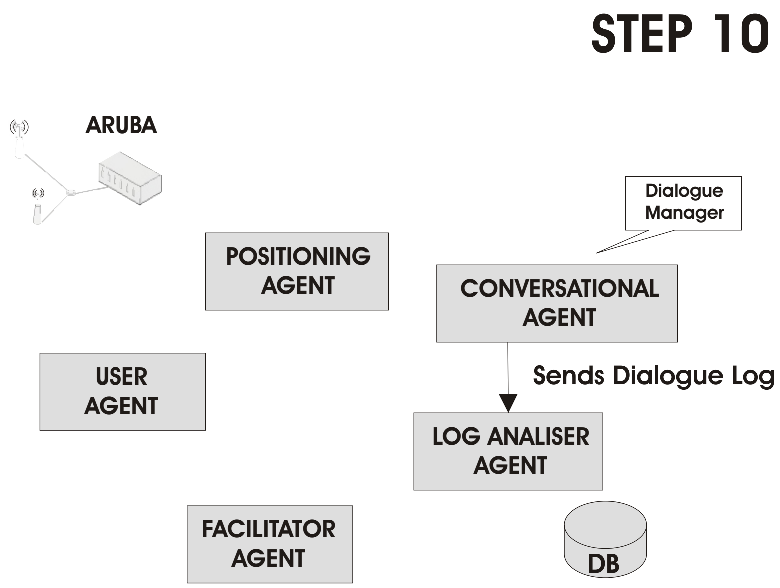


Figure 3.26: Step 10 — Agents interaction.

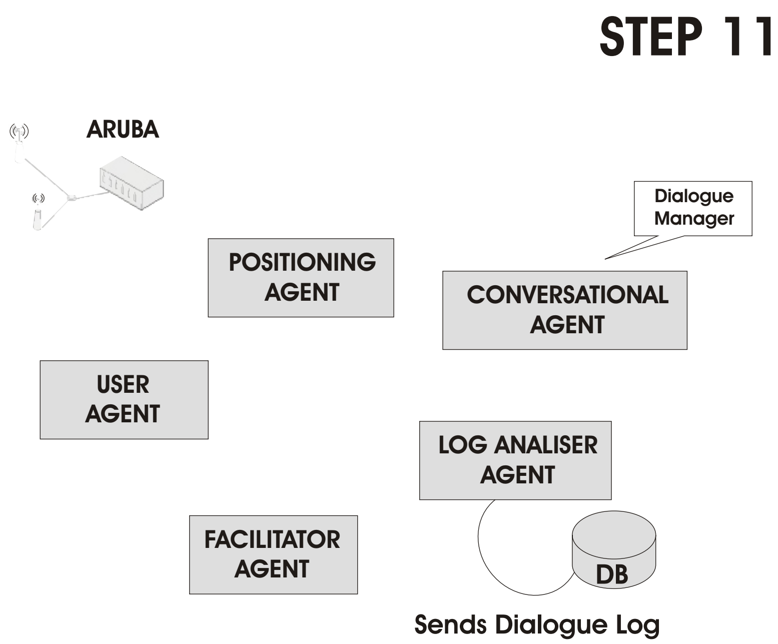


Figure 3.27: Step 11 — Agents interaction.

useful for the dialogue (ProviderAgent.QueryContext and UserAgent.ContextQueried) (see Figure 3.25).

11. The User Agent provides the context information that has been required (behaviours UserAgent.SendContext and ProviderAgent.ReceiveContext) (see Figure 3.26).
12. The Provider Agent internally invokes the Conversational Agent providing the context information about the user and the name of the file that will store the log file at the end of the current dialogue (behaviour ProviderAgent.InvokeDialogSystem).
13. The conversational agent manages the dialogue providing an adapted service by means of the context information that it has received.
14. Once the interaction with the Conversational Agent has finished, the Provider Agent reads the contents of the log file for the dialogue and send this information to the Log Analyser Agent (behaviours ProviderAgent.SendLog and LogAnalyzerAgent.ReceiveLog).
15. The Log Analyser Agent stores this log file and generates a user profile to personalize future services. This profile is sent to the Provider Agent (behaviours LogAnalyzerAgent.SendProfile and ProvderAgent.ReceiveProfile) (see Figure 3.27).

The free software JADE (Java Agent Development Framework) has been used for the implementation of our architecture. It was the most convenient option as it simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA specifications and through a set of graphical tools that supports the debugging and deployment phases. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. The synergy between the JADE platform and the LEAP libraries allows to obtain a FIPA-compliant agent platform with reduced footprint and compatibility with mobile Java environments down to J2ME-CLDC MIDP 1.0. The LEAP libraries have been developed with the collaboration of the LEAP project and can be downloaded as an add-on of JADE from this same website.

3.2.2.2 Positioning, facilitator and user agents

The main task of the Positioning Agents are users identification and location into the environment. To do this, these agents connect to the Aruba Positioning System (Sánchez-Pi et al., 2007) to read the positioning information. This information consists of an *x,y* coordinates and the building and floor information of the system. Facilitator Agents manage the different services provided by the system and reach an agreement with user agents and communicate the most suitable services to their preferences and context information. User Agents negotiate with Facilitator Agents, recommend services to other user agents, trust in other agents, and manage and improve an internal profile to receive adapted services according to it.

User agents main goals includes negotiation with the facilitator agent, recommend services to other user agents, trust in other agents, and manage and improve their internal profile to receive better services according to it. The information defined in the user profile stored in the user agent can be classified into two different groups:

- Personal information: user's name, gender, age, current language, skill level when interacting with dialogue systems, pathologies or speech disorders;
- User preferences: This set is split into different statistic groups, which are defined according to a specific domain-knowledge. This new subdivision allows the system to infer the preferences of each user regarding specific queries included in the task (i.e. in a railway information system to know that a specific user usually requires timetables information) or relative to specific values of attributes to be used in these queries (i.e. the preferences for travelling during a specific part of the day or by using a specific type of train).

Although other alternatives exist, JADE-LEAP agent platform was chosen since it is a Java-based and FIPA compliant agent platform, where agents communicate by sending FIPA ACL messages over a TCP/IP connection between different runtime environments on local servers or running on wireless devices. One local server hosts the positioning agent that rules over all the sensors of the domain (these sensors are just simulated inside the implementation of the central agent) that provide location information of users. Another server hosts the facilitator agent that matches the user profile with the services while one or more local servers host conversational agents acting on behalf of the different services. Each portable client device runs a JADE-LEAP container that hosts one single agent that is used to provide a way to interact with the user (through a GUI), so the user can be informed and interact with the other agents running on different servers. JADE/LEAP platform logically connects the agents on the different servers with each other and with the corresponding user agents running on the users' devices.

First we assume an initial minimal profile known by the client LEAP agent: name, agent role, passport number, nationality and travel info (flight numbers, companies, origin and target). This is what we consider the public profile that Client agent includes in the content of the first message exchanged with the central agent.

Other additional data can be included in the profile as: smoker/nonsmoker, languages spoken, number of times that the passenger visited cities of current travel, professional activity, travel motivation, relationship with other passengers, etc. Some of these would be considered private information therefore this private information would be just internally used inside Client agent to provide customized filter of the services offered by provider agents. The user decides whether he introduces these confidential data or not. Finally our intention is to include reputation information linked to agents, and specific services inside client profiles, but this is not already done. Once client agent in LEAP, provider and central agents in JADE were implemented, we have chosen specific cases studies to test such implementation in the AmI application domain.

3.2.3 BDI-modelling for AmI problems

The BDI Model (Belief–Desire–Intention model) was conceived as a theory of human practical reasoning (Bratman, 1987). The BDI model supports event-based reactive behaviour and proactive or goal–directed behaviour, so that is one of the reasons to choose this model for implementing the proposed context–aware multi–agent system. The main goal is to facilitate the utilization of mental concepts in the implementation, where this is regarded as appropriate by agent developer (Pokahr et al., 2003). This model is characterized by three types of attitudes: beliefs, desires and intentions. Beliefs are informative component of the system state (Rao & Georgeff, 1995b). Beliefs represent the information that an agent has about the world, and about its own internal state (Pokahr et al., 2003). Desire represents the motivational state of agents, which are captured in goals. It is necessary that the system also have information about the objectives to be accomplished or, more generally, what priorities are associated with the various current objectives (Rao & Georgeff, 1995b). Intentions capture the deliberative component of the system (Rao & Georgeff, 1995b). Plans, which are deliberative attitudes, are the means by which agents achieve their goals. A plan is not just a sequence of basic actions, but may also include sub–goals. Other plans are executed to achieve the sub–goals of a plan, thereby forming a hierarchy of plans. (Pokahr et al., 2003) The intention is to apply BDI model to the proposed context–aware multi–agent system. The design and implementation of each concept in BDI architecture as seen in Figure 3.28 (beliefs, goals and plans) will be described and in Section 3.2.4 the description of the complete architecture is modelled.

3.2.3.1 Beliefs

Each agent has a belief base to store the facts that make up the agents knowledge. These facts can be stored as Java objects in Jadex platform, and it is possible to capture de semantic of these objects using ontologies (Pokahr et al., 2003). In (Fuentes et al., 2006a) the proposal is heterogeneous domain ontology for the context-aware multi–agent system study in this research. A high level of abstraction for the ontology is defined, so that it covers dynamical and changing domains. The proposed meta–concepts include all context information of the multi–agent system, like location, participants (with their profiles), spatial region (buildings, rooms etc.), places or points of interest, services, products etc. In central agent a belief is used to represents the environmental knowledge, the user location and it knows provider services and location. In client agent, the initial belief is the user private profile.

3.2.3.2 Desires

According to (Pokahr et al., 2003), there are three kinds of goals: achieve, maintain and perform goal. They define, respectively, a desired target state, an agent keeps track of the state, and certain activities that have to be done. In this case, the goals are presented in a high level abstraction, so there is no distinction between the three types. Using methodology, agents' goals and functionalities are studied with more detail level in next section. The goals for each agent are described as follows. Central agent can reach the following goals:

- Detect Users: this goal implies that central agent has to locate and identify users.
- Register Users: central agent receives a request from users for registry, and send the respective agreement to users for register them. The goal of deregistering users is similar.

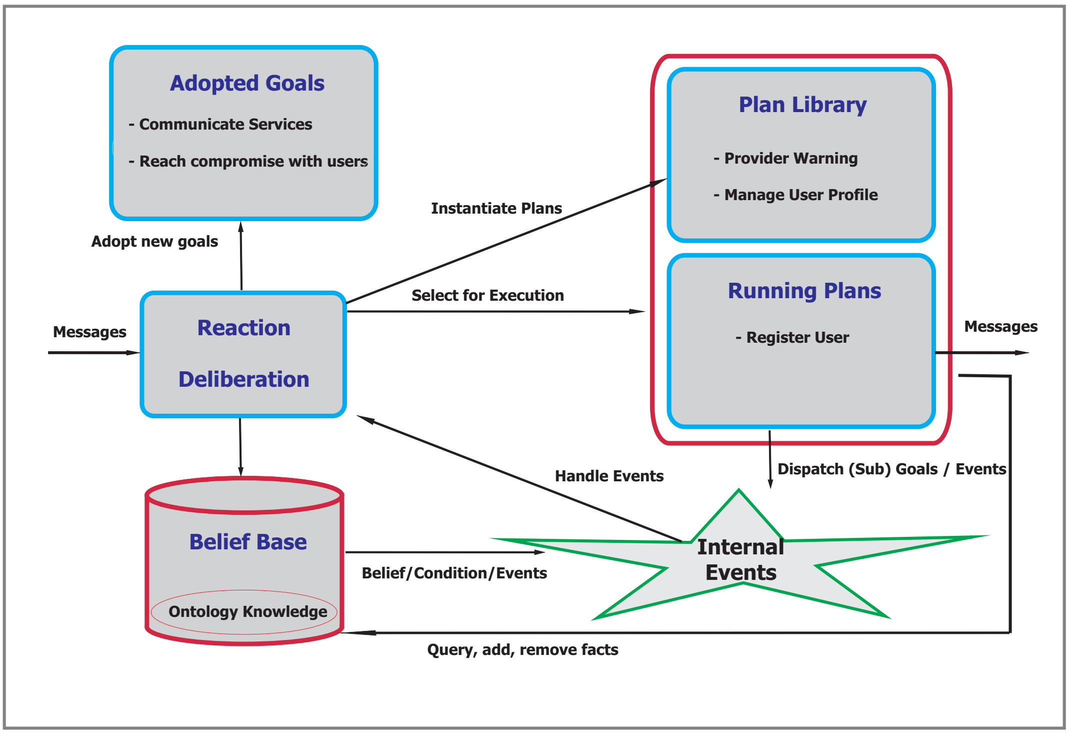


Figure 3.28: Example of BDI model application.

- Improve User Shared Profile: central agent receives user shared profile with the request to registry from user. Furthermore, central agent can update or improve user profile with some information like the time spent for users in visited places gathered by sensors etc.
- Filter Providers: central agent can filter providers by matching user profile with providers' information, and use the result of matching to warn closer providers. Goals in provider are closely related to dialogues between users and providers. Provider agent can achieve the following objectives:
- Communicate services: provider agents offer services to users according to the result of the matching realized by central agent.
- Reach a compromise with clients: provider agents negotiate with users and, after the negotiation phase, they can reach agreements and exchange information with them. User agents dialogue between them, but also they communicate with providers and central agent, so goals can be the following:
- Negotiation between users and providers: users can consult information by requesting to providers, and can receive services according to their profiles or location. Moreover, they can ask for agreements to providers and negotiate with them.
- Recommendation between users: users can recommend information to other users and they can also ask for recommendations.

- Trust in other agents: for sharing opinions with other agents and for improving the profiles with this information.
- Manage Internal Profile: users' agents can update their internal profiles with information about other user recommendations. Moreover they send a shared part of their profiles to central.

3.2.3.3 Intentions

The main functionalities of the multi–agent system are decomposed into separate plans. To achieve the goals, agents execute these plans. Initially, there are plans predefined in the plan library, and depending on the current situation, plans are selected in response to occurring events or goals. One main characteristic of BDI model is than the plans are selected in an automatic way: they can be executed from beliefs or form external messages that generate events (Pokahr et al., 2003). In the proposed context–aware multi–agent framework, plans represent the main activities of agents (Fuentes et al., 2006b). Central Agent plans can be described as follows:

- Register Users Plan: includes detection goal (locate and identify users) and register/deregister goal.
- Provider Warning Plan: this plan dispatched filter provider goal.
- Manage User Profiles: manage user shared profile goal Provider Agent Plans:
- Dialogue with users Plan includes two goals: offer services to users and negotiate with them for reaching agreements. User Agent Plans:
- Negotiation Plan: users negotiate with providers and make agreements with them.
- Dialogue between users: it includes recommendation goal and trusting goal.
- Manage Profile Plan: it concerns to manage internal profile goal.

3.2.4 Gaia Methodology

Gaia is a methodology for systems design based on agents which main goal is to obtain a system that maximizes some global quality measure. Gaia tries to help to go systematically from a few initial requirements to a design, so that the detail level is sufficient to be implemented directly. Gaia deals with both the macro (societal) level and the micro (agent) level aspects of design. It represents an advance over previous agent-oriented methodologies in that it is neutral with respect to both the target domain and the agent architecture (Wooldridge et al., 2000).

There are some previous research works that realize the analysis and design of agent systems according to Gaia methodology (Chiang et al., 2005; Ramanujam & Capretz, 2004; Tiba & Capretz, 2006). These researches present some differences with this proposal, in order to apply the current methodology.

One of these studies proposes a multi-agent system to configured autonomic services for telecommunication with multi-agent systems (Chiang et al., 2005). This proposal defines the system roles based in Gaia, and it focuses only in this phase of the methodology. Moreover, the reason for using Gaia methodology as a multi-agent system methodology is not clear. In contrast of this proposal, all the phases in Gaia are applied to our context-aware multi-agent system.

In (Ramanujam & Capretz, 2004) a multi-agent system for autonomous database administration is defined and the Gaia methodology is use to study the system analysis and design with best level detail as it is possible. The main difference with this research is the purpose of the multi-agent system, since the problem here is to analysis and design a context-aware multi-agent system for multiple applications.

In the case of SIGMA system (Tiba & Capretz, 2006), there is a combination of Gaia methodology with AUML methodology to provide more detailed description of the architecture. The application of Gaia is similar to our proposal but in this paper Gaia is combined with BDI model to get a better definition of the entities, characteristics, and functionalities of our multi-agent system. The goal is to extend with Gaia methodology the functionality obtains with BDI.

Applying Gaia to developing multi-agent systems, the analysis and design process consist in two phases: analysis phase, design phase (Wooldridge et al., 2000).

3.2.4.1 Analysis phase

The goal of this phase is to develop an understanding of the system and its structure. Gaia's view of agent system is like an artificial society or collection or roles, with interactions between them. A role is defined by four attributes: responsibilities, permissions, activities, and protocols (Wooldridge et al., 2000). This phase is concerned with the collection and organization of the system specification (Franco Zambonelli & Wooldridge, 2003), in particular:

- (i) the organizational structure)
- (ii) the environment model
- (iii) the preliminary role model
- (iv) the preliminary interaction model
- (v) organizational rules

The Environmental Model Since the importance of environments, because a multi-agent system is always situated in some domain, modelling this environment involves determining which entities and resources take part of the multi-agent system, for reaching the organizational goal (Franco Zambonelli & Wooldridge, 2003).

For the proposed context-aware multi-agent system with the main objective of adapting context information for providing customized services to users based on their location and profile, the environments is represented for multiple, heterogeneous and dynamical domains, as the behaviour of the system is always the same in any domain (an airport, a fairground, a shopping center, etc.) It provides the same functionalities but with different kind of information depends on the context. In (Fuentes et al., 2006a) there is an overview of how the environment for this context-aware system is represented by an ontology, since there is a need of define the context knowledge for the communication process between agents in the system.

The Organization Structure The organization structure defines the overall goals and behaviours of the system. As described in Section 3.2.3, there are three kinds of agents: central agent, provider agent and client agent that have to interact between them to reach the main goal of the system: adapt context-aware information to provide customized services to users. Each of them has to achieve specific goals in the organization and work together to reach the main objective of the system.

Role Model A role model identifies the main roles that agents can play in the system. A role is viewed as a description of the functionality of an agent, and it includes four attributes: permissions (resources use while performing a role), responsibilities (expected behaviours of the role), activities (actions without interaction between roles) and protocols (actions than involve interaction between roles). The following roles have been defined in the proposal multi-agent system:

- Provider Discover: it obtains closer providers and communicates with them to alert them for the presence of a user.
- User Manager: this role is responsible of coordinating all activities about users like locate, identify and register users, and improve their public profiles.
- Service Manager: it is necessary to manage the matching about services offered by providers and user profile.
- Service Provider: the main function is to communicate the offered services to users according to user location and profile. Another function is to compromise with clients for making agreements.
- Profile Manager: it deals with the update of internal user profile. This profile can be updated with information given by other users.
- Negotiate Role: users can negotiate with providers and reach agreements.
- Recommend role: users can communicate with other users to recommend places or points of interest, and, generally, all kind of information related to system services.

An example of the role model is describing as follows in Figure 3.29, according to Gaia specifications. It shows the roles corresponding to central agent, provider agent and client agent, with the permissions, responsibilities, protocols and activities (underlying) associated to them:

Role Schema: User Manager (UM)
Description This role is responsible for locating, identifying and registering users, as well as improving their public profiles.
Protocols and Activities Check-user-location, agree-registry, deregister-user, receive-registry-profile, register-user, identify-user, receive-user-sequence, improve-user-profile.
Permissions Reads user_location, user_profile, sequence Changes user_registry, user_profile.
Responsibilities Liveness: UM= (Check-user-location. Identify-user. Receive-registry-profile. Agree-registry. (Register-user | Deregister-user))n | (Receive-user-sequence. Improve-user-profile)n Safety: it is necessary to assure the connection with the location system and knowledge base

Role Schema: Service Manager (SM)
Description This role makes the matching about services offered by providers and user profile.
Protocols and Activities Match-services-profiles
Permissions Reads provider_service, user_profile Changes matching_result
Responsibilities Liveness: SM=(Match-services-profiles)n Safety: it is necessary the provider_service and user_profile is available to make the matching.

Role Schema: Provider Discovery (PD)
Description This role obtains closer providers and communicates with them to alert for the presence of a user.
Protocols and Activities Filter-provider, warn-provider
Permissions Reads matching_result Changes communication_information
Responsibilities Liveness: PD= (Filter-provider. Warn-provider)n Safety: if there is a success matching result, it is possible to communicate with the provider.

Role Schema: Service Provider (SP)
Description This role is responsible for informing about services and reach agreements with users after negotiation.
Protocols and Activities Offer-service, request-negotiation, agree-negotiation, exchange-information
Permissions Reads agree-negotiation, information_exchange Changes services_offered
Responsibilities Liveness: SD = (Offer-service)n | (Request-negotiation. Agree-negotiation. Exchange-information)n. (Offer-service) Safety: it is necessary to negotiate first for reaching an agreement and exchanging information.

Role Schema: Profile Manager (PM)
Description This role is responsible for updating internal user profiles and offers the possibility of sending a shared part to central agent
Protocols and Activities Update-internal-profile, send-shared-profile-registry
Permissions Reads user_profile, recommend_information Changes user_profile.
Responsibilities Liveness: PM= (Update-internal-profile)n | (send-shared-profile-registry)n Safety: it is necessary to receive external information, like recommend information to improve or update internal profile.

Role Schema: Negotiate Role (NR)
Description This role let agents negotiate and, according to this, receive new services or improved services.
Protocols and Activities Consult-information, receive-services, ask-for-agreements, receive-request-negotiation, exchange-information
Permissions Reads negotiate_information Changes services
Responsibilities Liveness: NR= (Consult-information)n | (Ask-for-agreements. Receive-Request-negotiations. (Exchange-information)n .Receive-services) n Safety: there is a negotiation phase for reach agreements and receive new services.

Role Schema: Recommended Role (RR)
Description This role offers the possibility to recommend information between users, and decides who to trust for sharing opinions.
Protocols and Activities Recommend, ask-for-recommendations, decide-to-trust, receive-recommendation
Permissions Reads recommendations Changes opinion_to_others, user_profile
Responsibilities Liveness: UM= (decide-to-trust. Recommend)n | (Ask-for-Recommendations. Receive-recommendation)n Safety: it is necessary to assure the connection with the location system and knowledge base.

Figure 3.29: Gaia role model for the context-aware multi-agent system

Interaction Model Interaction model is used to represent the dependencies and relationships between the roles in the multi-agent system, according to the protocol definitions. For the previous roles, several protocols are defined. They are represented in Figure 3.30.

The following Figure 3.31 shows a little example of how the interaction between the tree kinds of agents would be: Provider with Central agent, Central agent with Client (User) agent and Provider with Client agent.

Organizational Rules According to Gaia, the preliminary phases capture the basic system functionalities, characteristics and interactions in an independent way of the organizational structure. However, there are some relationships between roles and protocols which can be complemented with organizations rules for a best capture. These organization rules are responsibilities of the organization in a generic way (Franco Zambonelli & Wooldridge, 2003). In this case, organization rules correspond to the responsibilities analysed in role model. For showing this more clearly, here there are some examples of organizational rules defined about roles and protocols. For instance, there is a rule like the following:

$$(Filter - Provider)^n \rightarrow Warn - Provider$$

This means that Filter Provider must precede Warn-Provider and it can be executed n times. Each time Filter-Provider rule is executed, the action Warn-Provider will be executed after it. Otherwise, if we add the role, for instance Provider Discover role (PD) played by central agent, the organizational rule should be:

$$(Filter - Provider(PD))^n \rightarrow Warn - Provider(PD)$$

The rest of the organizational rules can be captured from the role model defined in Section 3.2.3 in the same way. In this paper we have considered not to extend more this section because it corresponds to the role model and focus in other important phases.

3.2.4.2 Design phase

This phase is composed by architectural and detailed design phase. The aim of designing is to transform the analysis models into a sufficiently how level of abstraction that traditional design techniques, included object-oriented techniques, may be applied in order to implement agents (Wooldridge et al., 2000).

1. Architectural phase It includes definition of the system's organizational structure in terms of its topology and the completion of the preliminary role, and interaction models.
2. Detailed phase It includes two models: (i) agent model and (ii) service model. They identify, respectively, the agent types in the system and the main services that are required to realize the agent's role (Franco Zambonelli & Wooldridge, 2003).

Receive-Registry-Profile		
Profile Manager	User Manager	Request registry
Receive a request of registry to the user with the shared part of the user profile.		Received Request and Profile

Agree-Registry		
User Managerr	Profile Manager	Confirmation message
Send a message to confirm the user registry.		User registered or not

Warn-provider		
Provider Discover	Service Provider	Alert message
Send a inform message to the closer provider role for alerting the presence of a user.		Received message

Offer-Service		
Service Provider	Negotiate Role	Offer Sent
Send offers about services to users, that can be negotiated.		Offer Received

Request-Negotiation		
Service Provider	Profile Manager	Negotiation required
Ask for open a negotiation process from provider to user..		Accept/Reject Negotiation

Agree-Negotiation		
Service Provider	Profile Manager	Accept Message
Send a message for accept negotiation.		Message Received

Exchange-Information		
Service Provider	Profile Manager	Object to exchange
Exchange some information for making agreements and improve offered services.		Improved offers

Send-shared-profile-registry		
Profile Manager	User Manager	Request registry
Send a request of registry to the user with the shared part of the user profile..		Accept/Reject request

Consult-information		
Negotiation Role	Service Provider	Object to consult
Users can consult information by asking to providers.		Consult

Receive-services		
Negotiation Role	Service Provider	Negotiate information
User Agent receives services according to their profiles and location, from provider agent		Received Services

Ask-for-agreements		
Negotiation Role	Service Provider	Request agreement
User agent can ask for agreements to provider, and negotiate conditions to receive services.		Accept/Reject agreement

Receive-Request-Negotiation		
Service Provider	Profile Manager	Negotiation request
User agent receive a request message for negotiation from provider		Accept/Reject Negotiation

Recommend		
Recommendation Role	Profile Manager	Object to recommend
User agent can recommend to other user agent some information about products, places, services etc.		Recommendation

Ask-for-recommendations		
Profile Manager	Recommendation Role	Request recommendation
User agents can ask to other agents for recommendations		Accept/Reject Recommendation

Receive-Recommendations		
Recommendation Role	Profile Manager	Recommendation
User agents can receive-recommendations from other user agents		Received Recommendation

Figure 3.30: Protocols for the context-aware multi-agent system.

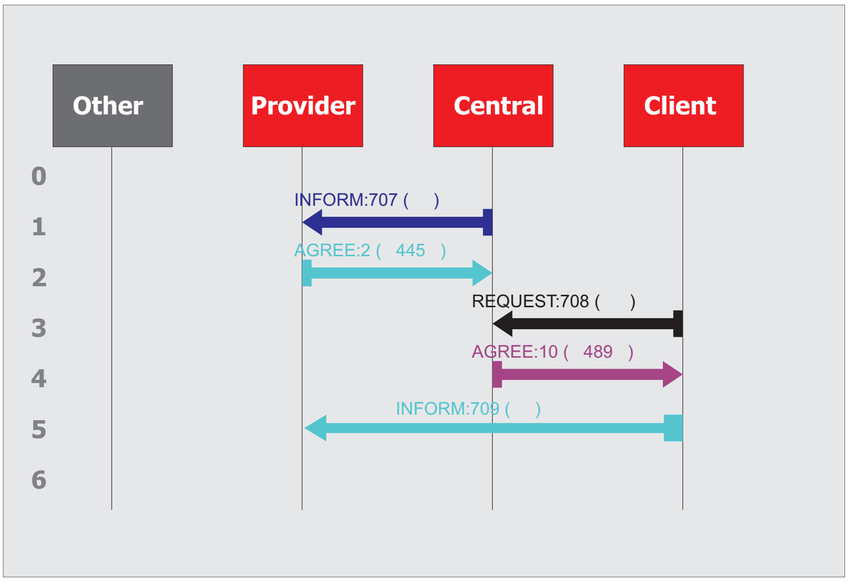


Figure 3.31: Jade agents interaction for the context-aware multi-agent system.

Agent Model According to Gaia, agents are entities that play different roles in the system. The definition of the agent model consist on identifying which specific roles play agents and how many instances of each agent have to be instantiated in the actual system. Agent model for the context-aware multi-agent system is shown in Figure 3.32.

Service Model The service model in Gaia methodology represents all protocols, activities, responsibilities and liveness, associate to the roles that agents plays in the system. This model is detailed as follows in Figure 3.33.

3.3 Multimodal user interface for AmI

Due to their rapid increase in performance and decrease in cost, computers have became an important part of our daily lives. We are surrounded by numerous electronic devices which provide information and functionalities; and increasingly we are interested in accessing them any time, anywhere and in our native languages. Thus, new interfaces are needed to provide natural, intuitive and efficient ways of communication between humans and computers.

As stated in the introduction, a conversational agent is a software that accepts natural language as input and generates natural language as output, engaging in a conversation with the user. To successfully manage the interaction with the users, conversational agents usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialogue management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules. Figure 3.36 shows a typical modular architecture of a conversational agent.

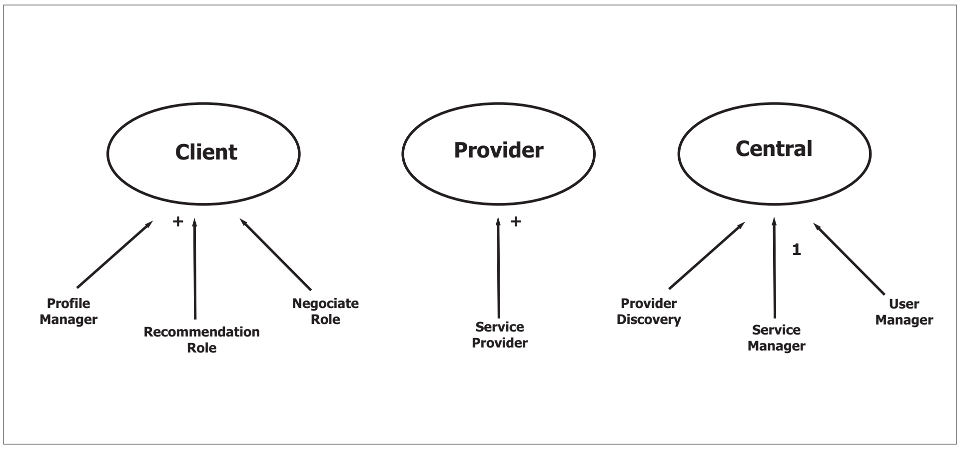


Figure 3.32: Gaia agent model for the context-aware multi–agent system problem.

Speech recognition is the process of obtaining the text string corresponding to an acoustic input. It is a very complex task as there is much variability in the input characteristics, which can differ depending on the linguistics of the utterance, the speaker, the interaction context and the transmission channel. Different applications demand different complexity of the speech recognizer. (Cole et al., 1997) identify eight parameters that allow an optimal tailoring of the speech recognizer: speech mode, speech style, dependency, vocabulary, language model, perplexity, SNR and transducer. Regarding the speech mode, speech recognizers can be classified into isolated-word or continuous-speech recognizers. Regarding the speech style, discourse can be read or spontaneous, the latter has peculiarities such as hesitations and repetitions that make it more complex to recognize.

There is not a universally agreed upon definition of the tasks that the dialogue management module has to carry. (Traum & Larsson, 2003) state that dialogue managing involves four main tasks: i) updating the dialogue context, ii) providing a context for interpretations, iii) coordinating other modules and iv) deciding the information to convey and when to do it. Thus, the dialogue manager has to deal with different sources of information such as the NLU results, database queries results, application domain knowledge, knowledge about the users and the previous dialogue history. Its complexity depends on the task and the dialogue flexibility and initiative. (Bernsen et al., 1994) provide a taxonomy which shows that for small and simple tasks single-word dialogue can be convenient with either system or user initiative and limited system feedback. However, for large and well-structured tasks, there is a need for system-directed dialogues with appropriate system feedback, tracking of the dialogue history and simple user models. For larger ill-structured tasks, mixed initiative dialogues are necessary, with dynamic predictions, linguistic and dialogue act dialogue history and advanced user modelling.

When it is necessary to execute and monitor operations in a dynamically changing application domain, an agent-based approach can be employed to develop the dialogue management module. The modular agent-based approach to dialogue management makes it possible to combine the benefits of different dialogue control models, such as finite-state based dialogue control and frame-based dialogue managing

Similarly, it can benefit from alternative dialogue management strategies, such as the

Services Schema: User Manager (UM)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Check-user-location	User location	Location checked	Users connected to wireless network	Users located
Identify-user	User location checked	Identification	Users located	The user is identified
Request-Registry-profile	User send a request message	Request for registry and user profile sent	Users connected to wireless network	Central Agent receive request from user
Agree-registry	Request for registry and user profile	Agree message and registry done	User send a request message	User Registered or not
Register-user	Proposed registry	Requested registry	Locate and identify users	User registered
Deregister-user	Proposed deregister or user out of the wireless network	Requested registry	Locate and identify users	User registered
Receive-user-sequence	External Information	Improved profile	Receive information from sensors, etc	The profile is improved
Improve-Profile	Information about user behaviour	Improved profile	Receive information about user behaviour	The profile is improved

Services Schema: Provider Discovery(PD)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Filter-providers	Result of matching	Closer provider	The result of matching is valid	Provider filtered by location and information
Warn-provider	Filtered Provider	Provider received alert message	Obtain closer provider	Provider informs closer users

Services Schema: Services Manager (SM)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Match-services-profiles	User profile known by central and provider information and location	Location checked	Users connected to wireless network	Users located

Services Schema: Profile Manager (PM)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Update-internal-profile	Other users recommendations	Profile updated	Received other users recommendations	The profile is updated
Send-shared-profile-registry	Need of registry	Send request message and profile to central agent	Obtain closer provider	Provider informs closer users

Services Schema: Recommend Role (RR)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Recommend-users	Proposed of recommendation	Accept recommendation or refusal	—	Recommendation succeed or failure
Ask-for-Recommendations	Need of recommendation	Request recommendations	—	User agent receive recommendations or not
Receive-recommendations	Request recommendations	Recommendations received	Asked for recommendations	Recommendations received by user agent
Decide-to-trust	—	Decision to trust	—	Share opinions with other agents

Services Schema: Services Provider(SP)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Offer-service	Provider alerted to inform	Services offered according to location and user profiles	Closer provider warning	User receive services about his preferences
Request-negotiation	Provider request negotiation to user	Process of negotiation is requested	—	User accepts or rejects the request message
Agree-negotiation	Request message for negotiation	Agree message for negotiation or nothing	Request message	Negotiation Process initiated
Exchange-information	Agree message for negotiation	Information for exchange between provider and user	Agree message	Negotiation process finished

Services Schema: Negotiate Role (NR)

Service	Inputs	Outputs	Pre-Condition	Post-Condition
Consult-Information	Propose of negotiation	Agreement or disagreement	—	Agreement reached or failure
Receive-services	Request for services or result of matching is valid	Services received	Matching done by central agent and provider is warned for offer services	Users obtain customized services and information
Ask-for-agreements	Need of agreements with other agent.	Request for reaching an agreement	—	Agreement accepted or rejected
Receive-Request-negotiation	User agent ask for agreement to provider	Process of negotiation requested	User Agent have request to reach an agreement	Negotiation process is open (or not) for exchanging required information
Exchange-information	Request for negotiation	Information for exchange between provider and user	Ask for agreements and request for negotiation	Negotiation process finished

Figure 3.33: Gaia service model for the context-aware multi-agent system.

system-initiative approach and the mixed-initiative approach (Walker et al., 1998), which can be used alternatively in an adaptive way. Furthermore, it makes it possible to combine rule-based and machine learning approaches.

Natural language generation is the process of obtaining texts in natural language from a non-linguistic representation. It is usually carried out in five steps: content organization, content distribution in sentences, lexicalization, generation of referential expressions and linguistic realization. It is important to obtain legible messages, optimizing the text using referring expressions and linking words and adapting the vocabulary and the complexity of the syntactic structures to the user's linguistic expertise. The simplest approach consists in using predefined text messages (e.g. error messages and warnings). Although intuitive, this approach completely lacks from any flexibility. The next level of sophistication is template-based generation, in which the same message structure is produced with slight alterations. The template approach is used mainly for multi-sentence generation, particularly in applications which texts are fairly regular in structure such as some business reports. Using this approach, it is possible to provide adapted system prompts that take into account context information.

Finally, text-to-speech synthesizers transform a text into an acoustic signal. A text-to-speech system is composed of two parts: a front-end and a back-end. The front-end carries out two major tasks. Firstly, it converts raw text containing symbols such as numbers and abbreviations into their equivalent words. Secondly, it assigns a phonetic transcriptions to each word, and divides and marks the text into prosodic units, i.e. phrases, clauses, and sentences. The back-end (often referred to as the synthesizer) converts the symbolic linguistic representation into sound. On the one hand, speech synthesis can be based in human speech production, this is the case of parametric synthesis which simulates the physiological parameters of the vocal tract, and formant-based synthesis which models the vibration of vocal chords.

On the other hand, concatenation synthesis employs pre-recorded units of human voice. Generally, it produces the most natural-sounding synthesized speech; however, differences between natural variations in speech and the nature of the automated techniques for segmenting the waveforms sometimes result in audible glitches in the output. Finally, HMM-based synthesis is a method in which the frequency spectrum (vocal tract), fundamental frequency (vocal source), and duration (prosody) of speech are modelled simultaneously by HMMs. Speech waveforms are generated from HMMs themselves based on the maximum likelihood criterion.

So, our intention was to provide the system with a dialogue system in the centralized solution, as well as in the distributed one. We will now explain how was this implemented.

3.3.1 Multimodal user interface for centralized system: Dialogue system

In order to provide with a human natural interaction we introduced a dialogue system in another layer in our centralized architecture. To ensure such a natural and intelligent interaction, it is necessary to provide an effective, easy, save and transparent interaction between the user and the system. This way, the need for better human-device interaction is clear. With this objective, in the last years there has been an increasing interest in simulating human-to-human communication, employing the so-called spoken dialogue systems (López-Cózar & Araki, 2005). Speech and language technologies allow users to communicate via one of the most natural, flexible, and efficient modalities: their own voice.

So, in order to facilitates developing, discovering, providing and accessing adaptable services through personalized speech-based interactions with a context aware services management engine, to the context-aware centralized architecture presented in Section 3.1 of this chapter we add another layer which represents the dialogue system. As can be observed in Figure 3.34, our architecture consists of three layers: i) sensing and rendering (sensors); ii) services manager (acquisition, representation and reasoning) and iii) dialogue system.

Sensing and rendering layer In the sensing and rendering layer there are a series of sensors which capture the users' activity (position, voice, environmental noise). Location information is acquired by the services manager layer which converts coordinates into zones, as well as the static user profile like: name, role, IP/MAC address, date/time. The rest of the profile information is then processed by the dialogue system layer (DS), which adapts its modules depending on this information. Then, depending on the interface decision, the services manager or the dialogue system finally decide which are the appropriate services to be provided. Context information is used and updated throughout the entire cycle.

Services manager We have implemented the Services Manager using the Appear IQ Platform (AIQ). The platform features a distributed modular architecture that, as can be observed in Figure 3.34, it is mainly composed of two parts: the *Appear Context Engine* (ACE) and the *Appear Client* (AC). The ACE implements a rules engine, where the domain-specific rules that we define determine what should be available to whom, and where and when it should be available. In our system, the context parameters defined for every device in our architecture include physical location, date/time, device type, network IP address, and user language.

The ACE is installed in a server, while the ACs are included in the user's devices. The network management is carried out by the *Appear Context Proxy* (ACP), which eliminates unnecessary traffic thus ensuring bandwidth for new user requests, and keeps a cache of active user sessions and most accessed services. When a wireless device enters the network, it immediately establishes the connection with a local proxy which evaluates the position of the client device and initiates a remote connection with the server. Once the client is in contact with the server, it provides the set of applications the user can access depending on his physical position.

Therefore the functionality of Appear depends mainly on the ACE. This module is divided into three modules that collaborate to implement a dynamic management system that allows the administrator to control the capability of each device once they are connected to the wireless network. They are: the *Push Module*, the *Device Management Module*, and the *Synchronization Module*. The *Push Module* manages the automatic distribution of contents to hand-held devices. It pushes services on these devices using client-side intelligence when it's necessary to install, configure and delete user services. The *Device Management Module* provides management tools to deploy control and maintain the set of mobile devices. Finally, the *Synchronization Module* manages file-based information between corporate systems and the mobile hand-held devices.

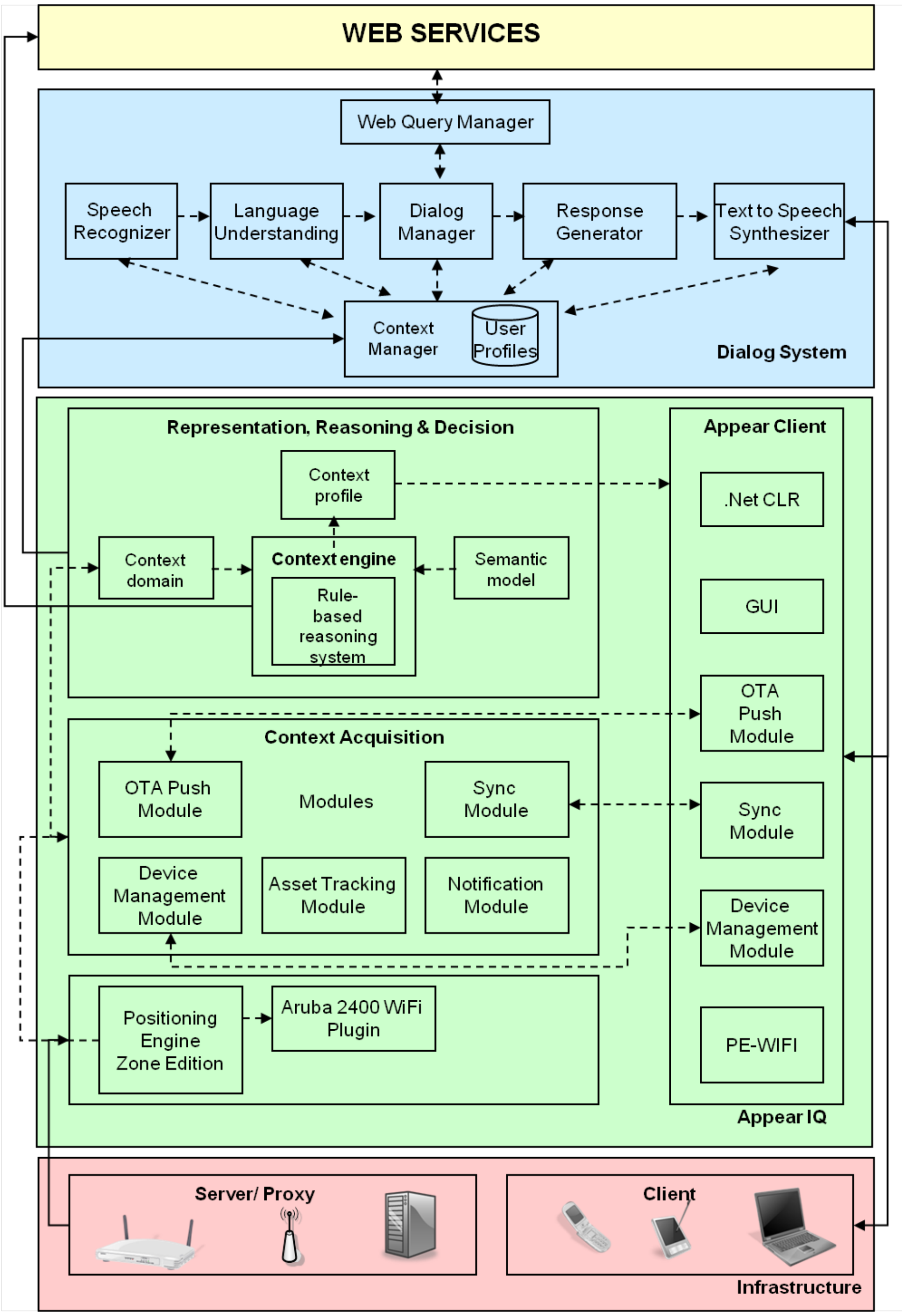


Figure 3.34: Centralized architecture using Dialogue System

Dialogue system The practical implementation that we propose for the dialogue system architecture is based on two main components: an IVR (interactive voice response) server and a set of web servers. The IVR provides users with multimodal web pages following the VoiceXML standard (World Wide Web Consortium (W3C), 2007b), the ASR and TTS interfaces, and VoIP and telephony technologies. Web servers connected to the IVR via Internet provide dialogue management facilities, grammars and system prompts, and the access to the different services provided by the system.

Context information is very valuable in order to enhance oral communication. For this reason, we have incorporated a *Context Manager* in our architecture. This module deals with creating, updating and loading the context information by means of specific profiles associated to each user. It also communicates this information to the DS modules during the interaction. The information of a user profile can be classified into three different groups:

- General user information. This set includes personal information, such as user's name, gender, age, current language, skill level when interacting with dialogue systems, pathologies or speech disorders, if any, and so on.
- General statistics. This second group comprises the number of dialogues and dialogue turns, their durations, the date of the last interaction with the system, etc.
- Usage statistics and user privileges. This set stores the counts of each action over the system that a user performs, and a mark of user clearance for each possible action. The set is split into different statistic groups, which are defined according to the specific domain-knowledge we have. This new subdivision allows the system to infer the preferences of each user among the different statistic groups, and among the different items belonging to the same group.

Static context is represented and managed in the AIQ layer using concepts like role, zone, and location. This context information is mapped into zones and times to get the right services. A mapping consists of a condition and an offering. The meaning of the mapping is that when a condition matches the context of the users' device, the offering will be used by that client. There are several setting values for the context elements that form the condition: Zone name, Time period, IP pattern, and Roles. The Zone name is known in the positioning system. Once the positioning system gives the location coordinates of the user's device, the plugging defined in Appear translates these coordinates and assigns a Zone name to them. The Time period makes the mapping valid only when the time period is valid. The IP pattern restricts the access to users whose IP addresses fulfill a certain pattern. The name of the offering that should be used when a condition matches is typed in the Offering field. The rest of features defined for representing the user profile are managed by the DS using the user profiles. The transmission of the context between the different modules is carried out by sending XML packages based on the OASIS Web Services Context Specification (Little et al., 2007).

The complete process followed by the system to adapt its behaviour taking into account contextual information is as follows:

1. Once the user's device (e.g. a PDA) is discovered by the WiFi sensor, a zone name is identified with his location and a set of categories of services are provided as icons on his GUI;

2. If the user decides to use the spoken communication interface to receive the information, immediately the ACE sends an XML package to the DS Context Manager informing about its identification and current location.

3. Using such information, the DS Context Manager selects the user profile of the recognized user and communicates this information to the different modules of the DS. Each module uses this information to load its specific information and models (e.g. acoustic models);

4. The user starts the interaction with the DS. Throughout the interaction, each module can update the active user profile. Depending on the information that is modified, the *Context Manager* sends the value of the new features only to the modules that requires such information;

5. At the end of the interaction, the user profile is updated using the information acquired during the last dialogue session.

3.3.2 Multimodal user interface for distributed system: Conversational Agent

Our agent-based architecture uses context-aware information to provide customized services to users by means of their interaction with conversational agents. As can be observed in Figure 3.35, the architecture consists of four different agent types that cooperate to provide a personalized service: Positioning Agents, User Agents, Facilitator Agents, and Conversational Agents.

Conversational agents, which marries agent capabilities with computational linguistics, have became a strong alternative to enhance computers with intelligent communicative capabilities, as speech is the most natural and flexible mean of communication among humans.

Once the conversational agent has recognized what the user uttered, it is necessary to understand what he said. Natural language processing is the process of obtaining the semantic of a text string. It generally involves morphological, lexical, syntactical, semantic, discourse and pragmatical knowledge. In a first stage lexical and morphological knowledge allow dividing the words in their constituents distinguishing lexemes and morphemes. Syntactic analysis yields a hierarchical structure of the sentences, however in spoken language frequently phrases are affected by the difficulties that are associated to the so-called disfluency phenomena: filled pauses, repetitions, syntactic incompleteness and repairs.

Semantic analysis extracts the meaning of a complex syntactic structure from the meaning of its constituents. In the pragmatic and discourse processing stage, the sentences are interpreted in the context of the whole dialogue, the main complexity of the stage is the resolution of anaphora, and ambiguities derived from phenomena such as irony, sarcasm or double entendre.

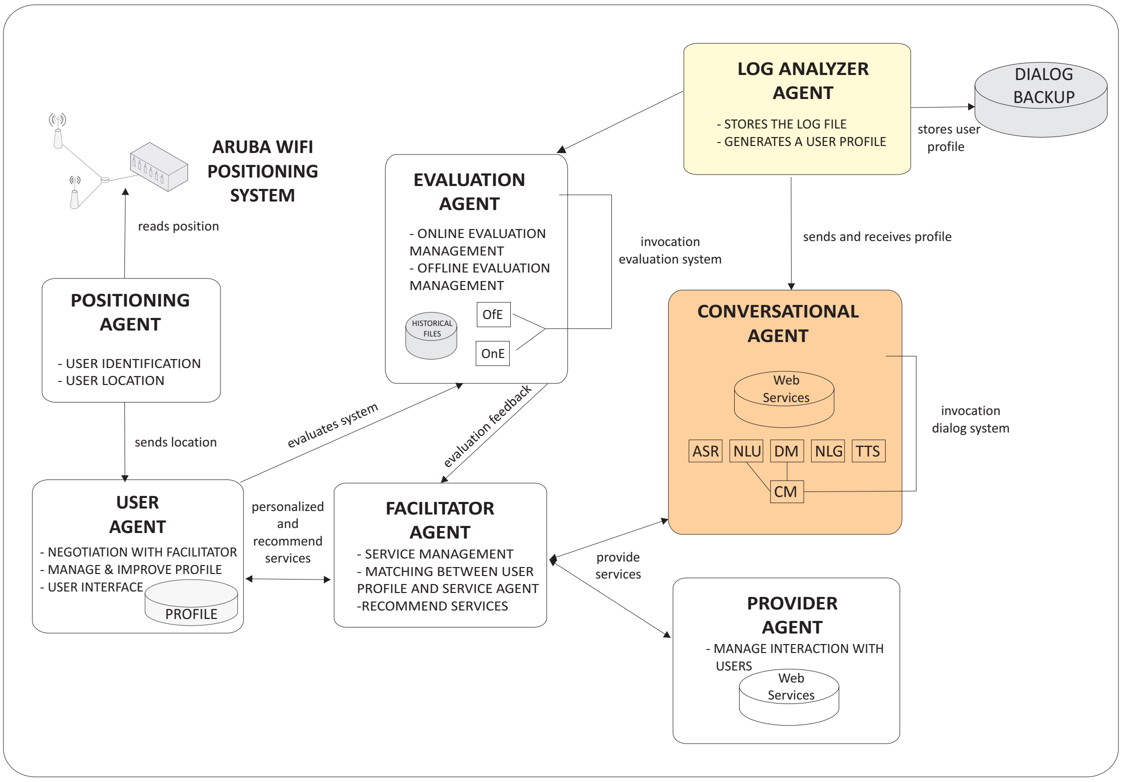


Figure 3.35: Distributed system architecture using a Conversational Agent.

Context information is very valuable in order to enhance oral communication. For this reason, we have incorporated a Context Manager in the architecture of the designed conversational agents, as shown in Figure 3.36. This module deals with loading the context information provided by the user and positioning agents, and communicates this information to its different modules during the interaction. However, the user is not required to transmit personal information or preferences, is only a suggestion to customize and shorten the dialogue. The agent representing the user communicates through messages the information the user wants to share. And finally the user does not need to respond to all suggestions made by the conversational agent.

To successfully manage the interaction with the users, conversational agents usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialogue management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules. Figure 3.36 shows a typical modular architecture of a conversational agent.

Speech recognition is the process of obtaining the text string corresponding to an acoustic input. It is a very complex task as there is much variability in the input characteristics. Once the conversational agent has recognized what the user uttered, it is necessary to understand what he said. Natural language processing is the process of obtaining the semantic of a text string.

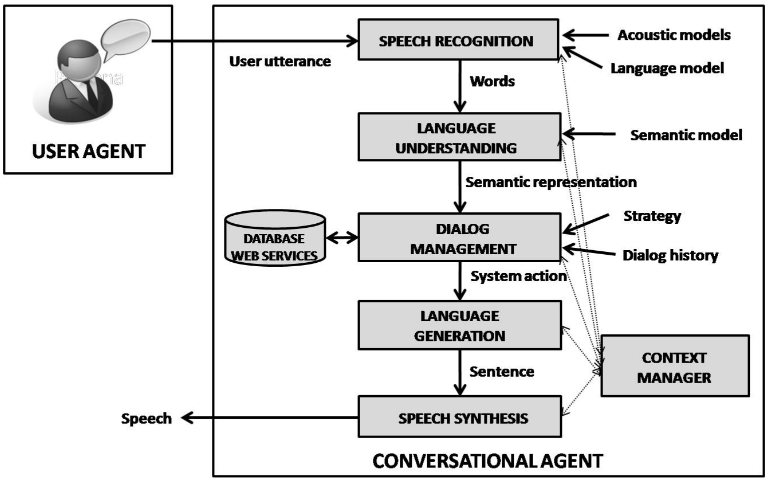


Figure 3.36: Schema of the architecture of a conversational agent.

It generally involves morphological, lexical, syntactical, semantic, discourse and pragmatical knowledge. The dialogue management module updates the dialogue context, provides a context for interpretations, coordinates the other modules and decides the information to convey and when to do it. Natural language generation is the process of obtaining texts in natural language from a non-linguistic representation. Finally, text-to-speech synthesizers transform the text into an acoustic signal (see Figure 3.35).

3.4 Evaluation in AmI

Evaluations methodologies allow researchers to assess the quality of the findings and to identify advantages and disadvantages of a system. The goal in evaluation of conventional systems is to proof that a system is more efficient. Normally, variables associated with efficiency are the time to complete a given task or the number of errors that have been made while fulfilling the task. However, in Ambient Intelligence when a system augments an environment enabling a user to do new things the metric is not straight forward any more. So, it is important before evaluating a context-aware system to figure out what is the evaluation goal. Context-aware services must dynamically adapt to the needs of the user and to the current physical, social and task context in which those needs are formed. Developing an effective context-aware adaptive service therefore requires extensive user-centred design. Currently, there is an increasing need for a generic evaluation framework of various technologies for context-aware systems. There are different types of quality approaches, like external quality, internal quality and quality in use but we focus in the quality in use measures in order to see the effects of its use in different context. In order to perform evaluation, it is necessary to measure the appropriate quality attributes and use the correct evaluation metrics depending on what is our evaluation goal. Following we present an evaluation methodology and in Chapter 4 we describe its application for each case study. Our intention is not to provide numeric measurements, but to describe an evaluation process for systems in AmI.

In AmI environments the evaluation process lays the following steps:

1. **STEP 1**- Definition of the purpose of the evaluation: It is where the set of inputs is defined. This step has tremendous importance since it is here where we represent the user needs, standards and the state-of-the-art technologies. This is the reference later to match against the main objectives and intentions of the evaluation. For instance, in AmI it not the same if we evaluate the impact of the user or the performance of the system. The two approaches are lay down in next sections.
2. **STEP 2**- Design: It concerns with the set up of the evaluation plan according to the previous step. Along with the creation of the appropriate tools taking into account for instance, to perform the evaluation in an automatic, objective and respectable way.
3. **STEP 3**- Execution: Involves the measurement of the previously selected characteristics, the comparison by using the selected criteria and the assessment of the results. This step is the one that provide the feedback to the developers to serve the subsequent iteration of the design process.

Once we have decided the purpose of the evaluation, we have designed a plan of evaluation, we need to decided which metrics involve. According the taxonomy presented in Chapter 2. Each system to be evaluated can be categorized based on one of the seven different criteria (as defined later in different subsections), and it will fall into one of the categories each time a different criterion is applied. The candidate parameters of interest for each system could be the union of the common parameters, the differentiating parameters and the key parameters associated with the category. Since there are multiple criteria employed in the taxonomy, any system can belong to multiple categories. Therefore, the set of categories the system belongs to can be used to define its character. In Section 3.4.1 we tackle evaluation of the impact of users and in Section 3.4.2, the evaluation of the performance of the system is described based on our distributed approach.

3.4.1 User-centred evaluation

There exist metrics for evaluation (scientific, component level, and impact) apply to AmI as detailed in Chapter 2. So for AmI scenarios we need to take into account specific metrics. We would want to address the recognition of a number of attractions and the ability to deliver the information corresponding to the attraction. i.e for railway information system we defined seven measures for the comparison of the dialogues acquired using or not context information; or we would like to know the satisfaction rating of the user with a particular activity or maybe we can measure the utility of the information delivered on attractions. So in the evaluation process we will focus on the application purpose criteria: assurance, assistive, return of investment, experience enhancement and exploration. The requirements and emphasis on various performance parameters are heavily dependent on their primary purposes.

Assistive systems The goal is to assist users with special needs to bridge the gap between the capabilities of able and disabled persons. Assistive services make available new opportunities to users by enhancing and expanding their communication, learning, participation, and achieving greater levels of independence, well-being, and quality of life (Helal et al., 2007). We found assistive systems in AAL, e-Heath, Transportation, etc. For these systems, the key parameters to be evaluated are their *usability*, *safety*, and *invisibility*.

Return on investment (ROI) systems ROI type of services serves primarily to increase the efficiency of users or the environments they are in, and potentially streamline the routine tasks and remove inefficiency. More prominent in business environment, like U-Commerce. Key parameters of these systems are *speed* and *efficiency*, especially of system response time and failure rate.

Assurance systems Assurance systems are designed primarily to provide a guarantee that proper actions will take place at all times, even under extraordinary and unexpected emergency situations. Fall detection system in AAL, e-Heath. Key factors in the evaluation of these systems are *safety* and *sentience*. It is critical for these systems to be reliable and secure, and to have a high level of fault tolerance and privacy preservation. *Quality of the context* is the key enabler to make sure that the systems can respond appropriately under the current context.

Experience enhancement systems Experience enhancement services focus on enhancing and enriching user experiences while interacting with pervasive computing systems. They provide additional or enhanced opportunities for learning, entertainment, or sensual experience. Existing experience enhancement systems includes scenarios where personalization take place. It can be, AAL, e-Health, Transportation, U-Commerce, Education When evaluating these systems, we look at *sentience* and *usability* in particular.

Usability and quality of context (QoC) are described as are the ones we will use later in Chapter 4.

3.4.1.1 Measuring usability

Metrics for usability are variables that are measurable in an objective manner. These variables are structured in three groups and we detail which one we use for this kind of scenario:

1. Effectiveness: Variables that allow us to measure the accuracy and completeness with which it achieves the objectives of a specific task. The most typical are:
    - Total percentage of completed tasks.
    - Percentage of task completed in the first attempt.
    - Percentage of users who complete the tasks.
    - Ratio of successes and failures.
    - Number of times users request help by not knowing what to do.
2. Efficiency: Refers to the effort that a user has to do to get a goal. Some typical variables are:
    - Time taken to complete each task.
    - Percentage or number of errors.
    - Percentage of errors or problems as its severity.
    - Time taken to recover from errors.

  - Number of clicks made in completing a task.
  - Number of pages visited to complete a task.
  - Time spent on specific pages or page groups.
  - Percentage or number of times you go for support, FAQ or similar.

3. Satisfaction: Refers to those who have more to do with the emotional or subjective. To measure the degree of satisfaction you can use criteria such as:

  - Percentage of users that after using the product would recommend to a friend.
  - Proportion of positive and negative adjectives that every user of the product.
  - Percentage of users who rate the product easier to use than any direct competition.
  - Number of times the user expresses satisfaction of dissatisfaction.

3.4.1.2 Measuring quality of context (QoC)

Quality of Context (QoC) was first defined in (Buchholz et al., 2003) as "any information that describes the quality of information that is used as context information". Later on, QoC has also been defined in (Krause & Hochstatter, 2005) as "any inherent information that describes context information and can be used to determine the worth of information for a specific application". Precision, probability of correctness, trust-worthiness, resolution, and up-to-dateness has been identified as important QoC parameters. This list has been extended to include accuracy, completeness, representation consistency, and access security selected on the basis of user's concern in the quality of context information.

Traditionally context-awareness is the ability of an application to adapt itself to the context of its users, whereby a user's context can be defined broadly as the circumstances or situations in which a computing task takes place (Meyer & Rakotonirainy, 2003). One of the most common contexts is the location of the user (or of objects of interest). For the scenarios we have implemented, location can be obtained using a variety of different alternative sensor types, WiFi, GPS, UWB, RFID-tags, bluetooth, video cameras and even body sensor networks. The quality (which is quantitatively application-specific) of the location information acquired by different sensors will however be different. For instance, ultrasonic badges can determine location with a precision of up to 3 cm, while RF lateration is limited to 1–3 m. Thus, we can define properties, which we call Quality of Context (QoC) attributes, that characterise the quality of the context data received.

While different types of contexts will have QoC attributes specific to them, there are certain attributes that will be common to most contexts and they are:

- Precision measures how accurately the context information describes reality, e.g. location precision.

- Probability of correctness (poc) measures the probability that a piece of context information is correct. For instance, determining the current posture of a person (sitting, standing, lying on the floor in distress) using a video camera has a different probability of correctness than using pressure sensors in the furniture. This metric differs from precision, as precision measures accuracy when a measurement is correct, whereas this metric describes the probability that the measurement is correct in the first place.

- Resolution denotes the granularity of information. This can for example mean spacial coverage. For instance, in a kitchen there can be hot spots with high temperature (oven, cooker, toaster), which may not be picked up by temperature sensors in the room if spacial coverage is low. Increasing the number of sensors in the room and optimising how they are spread can achieve finer spacial granularity.

- Up-to-dateness specifies the age of context information. Also of interest is how likely the measurement is still accurately describing the present.

- Refresh rate is related to up-to-dateness, and describes how often it is possible or desired to receive a new measurement. Different applications will have different refresh rate requirements. However, it is preferable to keep the refresh rate as low as possible whilst still delivering adequate context information, so as to minimise resources such as wireless network bandwidth whilst aiming to maximize the lifetime of battery- powered sensors.

In order to provide with a user-centred evaluation, we have developed a self reported data toolkit called My feedback, which will help to capture user's context, user's ratings, intentions and actions. These data can be acquired from an offline system or an online system that is installed in the user's device. Although we developed both, it is particularly useful and more accurate when collected during or shortly after key moments of interest while still fresh in the user's mind and they do not require retrieval or reconstruction data from memory but access to and accurate reporting of information available to conscious awareness. My feedback toolkit runs on SmartPhones, PocketPCs, TabletPCs, and desktop machines running Microsoft®Windows. Questionnaires are triggered based on the movements of users. To maximize user response, a numerical ratings (e.g., Likert scales) is employed because it is much more efficient. However, this efficiency is at a cost of losing qualitative data. Thus, in this study, we also use an open question where users can recommend the correct service he would like to receive at that time and at that place.

3.4.2 Multiagents system (MAS) evaluation

Evaluation performance of the system is a complex task but it is harder when we deal with distributed systems in AmI environments. First, additional parameters are necessary compared to centralized systems, for example: maintainability, security and privacy and it is necessary also to take into account the heterogeneity and diversification of distributed systems. The evaluation of MAS, as part of a distributed system, makes possible two main objectives. Firstly, to understand their behaviour and secondly, to compare the operation of several systems. In the literature there are works addressing this evaluation based on the architectural style (Davidsson et al., 2006); software engineering related criteria and characteristics of MAS (Mylopoulos et al., 2002); (Giunchiglia et al., 2002), or the complexity of interactions (Joumaa et al., 2008). The evaluation of the distributed nature of MAS and the complexity of the interaction inside them is a very difficult task. The consideration of the interaction as the most important characteristic of MAS (Wooldridge & Ciancarini, 2001), allows studying and comparing this kind of systems at the level of their interactions.

Here, we suggest an assignment of evaluation values to Agents interaction in an specific MAS architecture for providing context services. This evaluation is mainly based on the relevance of the messages content brought by an interaction. For dependant nature of the relevance of the messages, the valuation has to be adhoc, but here we provide an example of how interesting is this alternative in order to evaluate any MAS architecture theoretically.

The consideration of interaction as the main of evaluation has been addressed by other researches. These studies have verified that this kind of evaluation originates different types of problems (Joumaa et al., 2008). Firstly, the effect of an interaction unit (a single message) in an agent system could be equivalent to the definition of *n* units (messages) in another system. This way, the weight assigned to the same interaction is *1* in the first system and it *n* in the second. Secondly, the interaction units that are received and cannot be used by an agent could be a bias in the measurement of interaction in MAS.

Our proposal is based on (Joumaa et al., 2008). The first task is to classify the possible received messages into specific sets sharing the same type. Then, a weight is associated to a message according to its type. If two messages with the same type produce very different effects on the agent, then this assignment does not provide a correct solution. The effects of considering interactions as the main feature of the evaluation, consists of initially processing a message and then decide a responsive action. The initial processing is carried out in two phases, which consist of the memorization that deals with the change at the internal state caused by the received message, and the decision that concern the choice of the action that will be handled. According to the evaluation model, two kinds of functions are considered:

- A function *Interaction* associates a weight to each message according to its type. This function can be computed adopting the primitives proposed by (Gaspar, 1991) to the type of interaction. This work describes four possibilities kinds of messages: present, request, answer, and inform. These four types have to be distinguished due to the different basic behaviors that they model from the sender or the receiver points of view. Therefore, if M^A_{sent} : the set of messages sent by agent A and if $M^A_{received}$: the set of messages which may be received by agent A, the function Interaction associates for a message sent by the agent A, a message received by the agent B:

$$Interaction = M^A_{sent} \rightarrow M^B_{received} \quad (3.1)$$

- This solution partially resolves the problem. However, it is only valid if two messages of the same type have equivalent effects on the agent. This way, we introduce the function Φ (Joumaa et al., 2008) to associate a different weight to a message according to the change on the internal state and the actions triggered after its reception. This function evaluates the different effect of a message in agent systems. For better understanding Φ is divided into two terms: the first one evaluates Decision, DD^A and the second one evaluates Memorization, MM^A. The term MM^A associates a value to the variation of the internal state (caused by memorization step). To quantify these two terms, some measurable characteristics of the internal state must be defined. The specification of these characteristics is related to each specific application domain. The variation on one of these characteristics implies the function MM^A is considered as the sum of these weights:

$$MM^A = M^A_{received} \times S^A \to S^A \quad (3.2)$$

With regard to DD^A, this term associates a value to the triggered actions (i.e., results of decision step). To quantify this term, different types of actions must be defined and associate a weight to each of them. Then, the value of the function DD^A is calculated as the sum of the weights assigned to the triggered actions. Let S^A be the set of possible internal states for the agent A and let A^A be the set of actions may be done by agent A, then:

$$DD^A = M^A_{received} \times S^A \to A^A \quad (3.3)$$

Finally, the function Φ is defined as the sum of these functions DD^A and MM^A and the evaluation of the interactions in the MAS is based on the combination of the two functions Interaction + Φ.

3.4.2.1 Evaluation of our proposed MAS architecture

In this section we present the application of the evaluation method described in the previous section o our context-aware agent architecture. We compute the described evaluation functions and assign different weights to each message in the agents interaction. Now, we will define the Interaction function and also the Φ function.

Weights vs type of message: Function *Interaction* The Interaction function according to the four message types described in the previous section (present, request, answer, and inform). Following, we detail the different phases during the interaction of the different agents and the different messages that are generated for the provision of the service. Three types of messages have to be distinguished because of the different basic behaviours that they model from the sender or the receiver points of view:

- A request includes a change of state of the sender, waiting for the answer.
- An inform includes no change of state for both the sender and the receiver. It might generate other informs, and possibly answers.
- A present includes a possible change in the state of the sender and/or of the receiver. Typically, a present will enable entering a society and introduce itself to other agents

Services are offered by means of a connection with the conversational agent.

1. The ARUBA positioning system notices a change in the position of a given User agent: *no message involved*
2. The Positioning Agent reads the information about position provided by the ARUBA Positioning Agent in the corresponding file: *no message involved*

3. The Positioning Agent communicates the information about position (coordinates x and y) and place (Building and Floor) to the User Agent: *present message*.
4. Once a User Agent knows its location, it asks the Facilitator Agent about the different services that are available in that location : request message.
5. The Facilitator Agent informs the User Agent about the available services: *inform message*
6. The User Agent then decides the services in which it is interested: *no message involved*
7. Once the User Agent has selected a specific service, it communicates its decision to the Facilitator Agent and ask it about the service providers that are available: *request message*
8. The Facilitator Agent informs the User Agent about the identifier of the Conversational Agent that supplies the required service in the current location: *inform message*
9. The User Agent asks the Conversational Agent for the required service: *request message*
10. The Conversational Agent asks the User Agent about the context information that would be used to adapt the provided service: request message
11. The User Agent provides the context information that has been required: *inform message*
12. The conversational agent manages the dialogue providing an adapted service by means of the context information that has been received: *no message involved*
13. Once the interaction with the Conversational Agent has finished, this agent sends a log file to the Log Analyzer Agent: *inform message*
14. The Log Analyzer Agent stores this log file and generates a new user profile to personalize future services. This profile is sent to the Conversational Agent: *inform message*

Therefore we associate the next values to the message types:

- request: 2 (a change of state, a reaction produced)
- inform: 1 (no change of state)
- present: 1.5 (1 or 2 change of state)

In order to assign weights to each type of messages, we follow this criteria: if there is any interest over a negotiation from the Users side, then the maximum weight is assigned to the Agree-proposal or Reject-proposal messages; if it is the system who tries to recommend a service based on its behavior, the maximum weight is assigned to Inform-ref or Propose messages.

Weights vs treatment of a message: Function Φ As stated in earlier, the Φ function computes the variation of the internal state of the agent caused by a memorization step and also a decision step. Memorization is evaluated by means of the MM^A. The defined ontology allows to measure that the internal state has changed due to the number of concepts involved or the number of attributes involved. We could have also considered the different relevance of the set of attributes and concepts by assigning different weights to each of them. For example:

- Phase 10: Changes in concepts and their attributes may have a low value. In this phase, the User Agent provides the context information. Since the User Agent is never forced to transmit its personal information and preferences, the relevance of this package is low. The weights that are assigned to the context information depend on the utility of concepts and attributes to achieve the provision of the required service during the dialogue interaction.
- Phase 3 and 12: The weights assigned to the changes produced by the messages in the phase 3 depend on the number of services that whether available (positive) or not (negative) in the specific location (position and place). In this phase, the Positioning Agent communicates the position and place information to the User Agent. The weights are assigned similarly to the changes produced by each message in the phase 12. In this phase, the Facilitator Agent informs the User Agent about the services that are available in the current position.
- Phase 13: The utility of the messages corresponding to this phase, in which the Conversational Agent sends the log file of the dialogue to the Log Analyzer Agent, depends on the coincidence of this log with regard to the previous dialogues (partly time-decrescent function).
- Phase 14: Finally, a high weight is assigned to the message of this phase, in which the Log Analyzer Agent sends a user profile to the Conversational Agent. Since this profile is generated using several dialogue logs to personalize the provided service, a high relevance is assigned to this message.

The term DD^A indicates the variation of the internal state due to a decision step. This function associates a value to each triggered action. Different types of actions must be defined to quantify this term, each having a specific weight. Then, the value of DD^A is calculated as the sum of the weights of the triggered actions. The set of actions involved in our agent architecture can be classified as external or internal. External actions involve communicative responses for the given message. The weight of this reactive action is equivalent to the weight of the content included in the received message. Internal actions involve the processing and decision making described in the following phases:

- Phase 4: Once a User Agent knows its location, it asks the Facilitator Agent about the different services available in that: request message. Simple query to the internal database of the Facilitator Agent. No intelligence involved: *minimal weight.*
- Phase 6: The User Agent decides the services in which it is interested. Intelligent and relevant decision with a real economic cost: *maximal weight.*

- Phase 8: The Facilitator Agent informs the User Agent about the identifier of the Conversational Agent that supplies the required service in the current location: Simple query to the internal database of Facilitator agent. No intelligence involved: *minimal weight*.
- Phase 11: The User Agent provides the context information that has been required. Intelligent and relevant decision with a privacy cost: *maximal weight*.
- Phase 13: Once the interaction with the Conversational Agent has finished, the Conversational Agent sends the log file generated after the dialogue to the Log Analyzer Agent. Simple query to the internal database of Conversational agent. No intelligence involved: *minimal weight*.
- Phase 14: The Log Analyzer Agent stores the log file and updates the user profile to personalize future services. Intelligent and relevant decision: *medium weight*.

Then we need to compute DD^A function that associate the variation of internal state caused by decision step. This function associates a value to the triggered actions. To quantify, certain type of actions must be defined. A type of actions having a weight. Then, the value of the function DD^A is considered as the sum of the weights of triggered actions. The set of actions involved in our agent system can be classified as external and internal. Where the external actions means communicative responses to the given message, where the weight of this reactive action is equivalent to the weight of the content included in the responsive message. On the other hand, internal actions involve the processing and decision making of the next phases:

3.4.2.2 Example of application: Airport domain

To introduce an experimentation case we use two different architectures. We compare them using different types of messages for a common domain: Airport presented in (Sánchez-Pi et al., 2007). In previous work we design an architecture that has three types of agents: Central Agent, Provider Agents and User Agents (Fuentes et al., 2006b). From now on we call it MAS-CENTRAL-AGENT. Later we adapt this architecture to cope with new functionalities including the speech based interface. From now on we call it: MAS-CONVERSATIONAL-AGENT. New agent's functionalities in this new architecture state as follow: *Positioning agent* main tasks rely on the user identification and user location into the environment. *Facilitator agent* is the responsible of the services management and the discovering of services agent identification. *Conversational Agents* provide the specific services. Finally, a *Log Analyzer Agent* generates user profiles that are used by Conversational Agents to adapt their behaviour taking into account the preferences detected in the users' previous dialogues.

If we compute the amount of messages exchanged of each type for each architecture, we could then draw some conclusions. We take experimentation using "Service Recommendation" in an Airport Domain (Sánchez-Pi et al., 2007) for MAS-CENTRAL-AGENT and MAS-CONVERSATIONAL-AGENT. The comparison involves the following five type of messages: *Agree-proposal; Inform-ref; Propose; Query-if; Reject-proposal; Request*. For instance, once the passenger John Mayer is inside the Airport, the system recommends him with a SPA service based on the reputation. An example of message in FIPA is:

```
(inform-ref
:sender (agent-identifier :name SPA)
:receiver (set (agent-identifier :name john mayer))
:content
((action (agent-identifier :name SPA)
(try (spa $product $reputation))
:protocol fipa-request
:language FIPA-SL
:ontology airport-ontology
:reply-with try-spa)
```

In order to assign weights to each type of messages, we follow this criteria: if there is any interest over a negotiation from the Users side, then the maximum weight is assigned to the Agree-proposal or Reject-proposal messages; if it is the system who tries to recommend a service based on its behavior, the maximum weight is assigned to Inform-ref or Propose messages. In the case of MAS-CONVERSATIONAL-AGENT, the Inform-ref message belongs to Phase 12 of our MAS interaction and would have a maximal weight because it is the system's intention to recommend a service based on the behaviour of the passenger. In this case is that the passenger is tired. Then, following the evaluation method described above and used in (Joumaa et al., 2008), MAS-CONVERSATIONAL-AGENT in contrast to MAS-CENTRAL-AGENT. Among the MAS-CENTRAL-AGENT properties the reactivity, communication, robustness and scalability highlight apart from the others. On the other hand, the MAS1 is more robust when evaluating fairness and load balancing. In this sense, the difference between the two systems, when performing the comparison on the number of messages sent is minimal. Therefore, MAS-CONVERSATIONAL-AGENT is valid.

4

Application Scenarios: Case Studies

Due to advances in communication technologies, AmI is increasingly entering in every aspects of our life and sectors, opening a world of unprecedented scenarios where users interact with electronic devices embedded in environments that are sensitive to the presence of users (Lyytinen et al., 2004). Tourism, healthcare, education, transportation, etc., are some of the environments where been developed ubiquitous systems and applications above AmI vision. Our systems allow to face problems directly and real-time in the field. This system is designed for normal people so by consequence contexts, phenomena and situations that can be studied, monitored and managed by the system are numerous and diverse.

A complete and detailed dissertation about the application fields of the systems lies inside the aim of this section. So, examples will be presented here, in order to underline the versatility of the system in different domains: transportation, e-health, ambient assisted living (AAL), education and business. We start from our own ontology, detailed in Chapter 3 and it is particularized later for every specific domains implemented with a distributed architecture. Figure 4.1 describe a summary of the scenarios and its particular implementation.

4.1 Transportation domain: Airport information system (Sánchez-Pi et al., 2007)

In last sections we explained how Appear represents the context and how the KBS for the airport domain was defined. Let us now move on to the configuration of the context using

SCENARIO	IMPLEMENTATION
AIRPORT DOMAIN	CENTRALIZED & DISTRIBUTED
RAILWAY DOMAIN	DISTRIBUTED + DIALOGUE SYSTEM
U-HEALTH	DISTRIBUTED
AAL	CENTRALIZED
UNIVERSITY	DISTRIBUTED
U-COMMERCE	DISTRIBUTED

Figure 4.1: Scenarios implemented.

Appear (Sánchez-Pi et al., 2007). See Figure 4.2. The objective of this initiative is to develop an electronic space in which different users in an airport can collaborate without restraint. Therefore the principal objective is to achieve a network centric environment that has the following characteristics:

- Open: any entity can enter and leave without encountering any barriers.
- Network-centric: Resources and competencies scattered throughout Europe can come together to enable collaborative value-creation relationships develop.
- No constraints: IT vendors, amongst others, cannot impose artificial technological constraints or barriers.
- Sustainable: Resources are limited and should be combined and used in a cost effective manner.

In light of this, two scenarios are now outlined that present a vision of how airport domain, based on an AmI metaphor, can serve both passenger and pilot as they proceed with their respective activities.

4.1.1 Overview of the airport information system

The Airport Information System (AIS) was developed to provide with useful information to users of Barajas Airport: passengers and personal crew of IBERIA Airline (airline captain, pilots, flight engineer, cabin crew, airline ground staff:boarding, airline ground staff: incidents, airline ground staff: check-in and supervisor). When they enter in the wireless network, the position of the client is evaluated and the systems initiates the negotiation of the set of applications the user can access depending on his physical position and preferences. To clarify the system's functionalities, following there are some of the scenarios where this Airport Information System can be found beneficial:

Scenario 1 — Airport/ Passenger

Don and Donna are two passengers about to take a flight to Amsterdam. They are arriving late to the airport, they have just 10 minutes left to make the check-in. They previously configure their preferences (window/aisle seat, vegetarian/normal food, special needs....). So once they arrive to the airport, both immediately connect their mobile devices to the airport wireless network and receives a reminder to make the check-in and also indications with the number and how to get to the check-in desk. Once they have their boarding pass, they enter Customs and the systems provides them with Identity and News services, Don likes to read "Times" and Donna "Washington Post", so system recognizes their preferences and provides the correct service to each user.

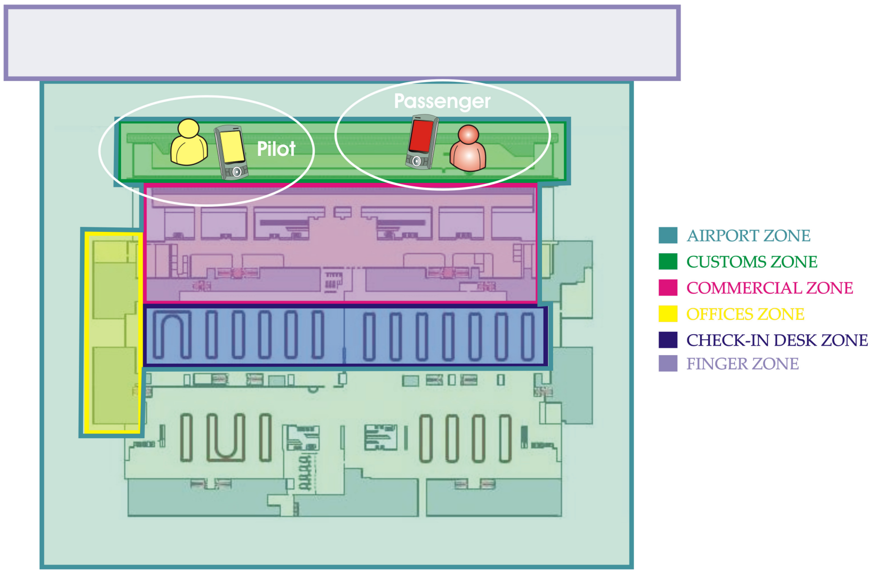


Figure 4.2: Airport Madrid Barajas

Scenario 2 — Pilot/ Airport

John is a pilot of the aircraft. Before to get to the airport he usually needs to go to the Airline Iberia Office first in order to get the flight documents but he did not have time today, so he enters the airport wireless network and the system sends him the Flight Plan, the Alternative Airports document, the Flight license and the Crew license. When he is arriving to Customs the system provides him with the Identity service and as the smoking zone is close to him, system fires alerts.

The two scenarios just described offer a brief but succinct illustration of how AmI could provide a transparent and seamless environment for solutions in an airport. Having reflected on the nature of an AmI environment, it is now necessary to consider what role the AmI environment should engage in when delivering services. In this case, the system will behave in a proactive manner. The system can proactively seek to match user's requirements and services.

4.1.2 Reasoning engine

For this case scenario we made two implementations, one using our centralized system and another one using the multiagent system.

4.1.2.1 Centralized approach

For the airport centralized approach, we configure context parameters in Appear, as presented in Section 3.1.

The definition of ROLES for our Airport domain example and Airline staff is:

- **ROLE Pilot**: Airline Staff in flight. Captain of the Aircraft and Pilots.
- **ROLE FTO**: Airline Staff in flight. Flight Technical Officer.
- **ROLE CBC**: Airline Staff in flight. Cabin Crew. They are stewardess and flight assistants.
- **ROLE ASCH**: Airline Staff in ground. Check-In. Airline Staff in check-in counters are in charge of assigning seats to passengers, also to identify their bags and to deliver them the boarding pass.
- **ROLE ASSV**: Airline Staff in ground. Supervisor. It is the person in charge of process the documentation for the correct flight dispatch, the one from the company as well as the related with the official policies that are part of it.
- **ROLE ASGB**: Airline Staff in ground. Boarding. These persons are placed in the boarding gates. They lead passengers to the aircraft after check that it is the flight where they should travel.
- **ROLE ASIC**: Airline Staff in ground. Incidence. They solved any doubt or incident passengers could have inside the airport. They also assist children traveling without companion, or handicapped persons who need special attention for travelling in a plain.
- **ROLE PASSENGER**: Passenger.

Context information in the system is used throughout the entire life-cycle of the service: selection based on the context profile, filtering of individual services, and enhancement of services at boot or runtime and the constant feed of context information to services during execution to allow service adaptation. For our example we developed some generic predicates for Roles and Zones in order to be clearer. There are one Predicates for each Roles and they are: PILOT, FTO, CBC, ASCH, ASSV, ASBG, ASIC and PASSENGER. The same happens with the Predicates for Zones. There is one for each zone: Check-In Desk, Offices, Commercial, Customs, Finger, and Airport.

Predicates for Roles are:

- **Predicate: PILOT** Condition: ROLE NAMES contains ROLE Pilot
- **Predicate: FTO** Condition: ROLE NAMES contains ROLE FTO
- **Predicate: CBC** Condition: ROLE NAMES contains ROLE CBC
- **Predicate: ASCH** Condition: ROLE NAMES contains ROLE ASCH
- **Predicate: ASSV** Condition: ROLE NAMES contains ROLE ASSV
- **Predicate: ASBG** Condition: ROLE NAMES contains ROLE ASBG
- **Predicate: ASIC** Condition: ROLE NAMES contains ROLE ASIC
- **Predicate: PASSENGER** Condition: ROLE NAMES contains ROLE Passenger

Predicates for Zones are:

- **Predicate: zone_check-in_desk** Condition: ZONE is Check-In_Desk
- **Predicate: no_finger** Condition: ZONE is not Finger
- **Predicate: zone_finger** Condition: ZONE is Finger
- **Predicate: zone_offices** Condition: ZONE is Offices
- **Predicate: zone_commercial** Condition: ZONE is Commercial
- **Predicate: zone_customs** Condition: ZONE is Customs
- **Predicate: zone_airport** Condition: ZONE is Airport

This is an example of the rules definition and its evaluation for providing the SERVICE_NEWS to a Passenger user into the Customs Zone.

Figure 4.3 shows the way Conditions and Predicates are evaluated for the Customs Zone and how the Offering Mapping takes place in order to offer different services to each User with an assigned Role in the system. It happens for every zone for which we will have one table like this one.

4.1.2.2 Distributed approach

During the context configuration procedure, we need to define every concept in our domain knowledge and represented in the ontology described in Figure 4.4. Some of the concepts adapted in the generic ontology are:

- Location: XCoordinate int; YCoordinate int
- Place: Building ID int; Floor Level int
- Service: Name String
- Product: Name String; Characteristics: List of Feature
- Feature: Name String; Value String
- ProfileDescription: Name; Characteristics: List of String
- ProfileData: Name; Characteristics: List of Feature

The current proposal includes agents implemented in JADE and LEAP that will make use of the user profiles to customize and recommend different services to other agents avoiding an obtrusive participation of a central server. These agents use a particular instantiation of a previously-defined generic ontology to represent the airport context including the buildings, rooms, zones, etc. . . , where users are moving inside with their mobile devices, and several points of interest which provide services to users.

OFFERINGS	CATEGORY	SERVICES	TYPE	EVALUATED PREDICATES				OFFERED TO
CUSTOMS	CONTROL	PASSENGER	XLS	passengers_control				PILOT
	REPORTS	CONTROL			p1_passengers_control is true			FTO
						zones_p1_passengers_control is true		CBC
							zone_customs is true	ASCH
						OR	zone_commercial is true	ASSV
					AND	roles_p1_passengers_control is true		
							pilot is true	
						OR	fto is true	
						OR	CBC is true	
				OR	p2_passengers_control is true			
						zones_p2_passengers_control is true		
							zone_check-in_desk is true	
						OR	zone_offices is true	
						OR	zone_airport is true	
					AND	roles_p2_passenger_control is true		
							pilot is true	
						OR	fto is true	
						OR	CBC is true	
						OR	ASCH is true	
						OR	ASSV is true	
		LOAD	XLS	load_control				PILOT
		CONTROL			p1_load_control is true			FTO
						zones_p1_load_control is true		ASCH
							zone_customs is true	ASSV
						OR	zone_commercial is true	
					AND	roles_p1_load_control is true		
							pilot is true	
						OR	FTO is true	
				OR	p2_load_control is true			
						zones_p2_load_control is true		
							zone_check-in_desk is true	
						OR	zone_offices is true	
						OR	zone_airport is true	
					AND	roles_p2_load_control is true		
							pilot is true	
						OR	FTO is true	
						OR	ASCH is true	
						OR	ASSV is true	
		SUPPORT		support_control				PILOT
		CONTROL			roles_support_control is true			FTO
						pilot is true		
					OR	FTO is true		
				AND	no_finger is true			
	NEWS	SPANISH	URL	news				PASSENGER
		NEWS			p1_news is true			PILOT
						roles_p1_news is true		
							pilot is true	
						OR	passenger is true	
					AND	zones_p1_news is true		
							zone_commercial is true	
						OR	zone_airport is true	
				OR	p2_news is true			
						passenger is true		
					AND	zones_p2_news is true		
							zone_check-in_desk is true	
						OR	zone_offices is true	
						OR	zone_custums is true	
		ENGLIISH	URL	news				PASSENGER
		NEWS			p1_news is true			PILOT
						roles_p1_news is true		
							pilot is true	
						OR	passenger is true	
					AND	zones_p1_news is true		
							zone_commercial is true	
						OR	zone_airport is true	
				OR	p2_news is true			
						passenger is true		
					AND	zones_p2_news is true		
							zone_check-in_desk is true	
						OR	zone_offices is true	
						OR	zone_custums is true	
		FRENCH	URL	news				PASSENGER
		NEWS			p1_news is true			PILOT
						roles_p1_news is true		
							pilot is true	
						OR	passenger is true	
					AND	zones_p1_news is true		
							zone_commercial is true	
						OR	zone_airport is true	
				OR	p2_news is true			
						passenger is true		
					AND	zones_p2_news is true		
							zone_check-in_desk is true	
						OR	zone_offices is true	
						OR	zone_custums is true	
	FLIGHT	ALTERNATIVE	DOC	alternative_arpts				PILOT
	RESOURCES	AIRPORTS		(	roles_alternative_arpts is true			FTO
					(	pilot is true		
					OR	FTO is true)		
				AND	no_finger is true)			
		FLIGHT	DOC	flight_plan				PILOT
		PLAN		(	roles_flight_plan is true			FTO
					(	pilot is true		
						FTO is true)		
				AND	no_finger is true)			
		CREW	DOC	crew_licence				PILOT
		LICENCE		(	piloto is true			
				AND	no_finger is true)			
		FLIGHT	DOC	flight_over_licence				PILOT
		OVER		(	pilot is true			
		LICENCE		AND	no_finger is true)			
		DEMO	PPT	demo_iinf_security				CBC
		INF SECURITY		(	p1_inf_security is true			
					(	cbc is true		
					AND	no_finger is true		
				OR	p2_inf_security is true)			
						roles_p2_inf_security is true		
							pilot is true	
						OR	passenger is true	
					AND	zone_finger is true		

Figure 4.3: Evaluation of Conditions and Predicates for the Customs Zone

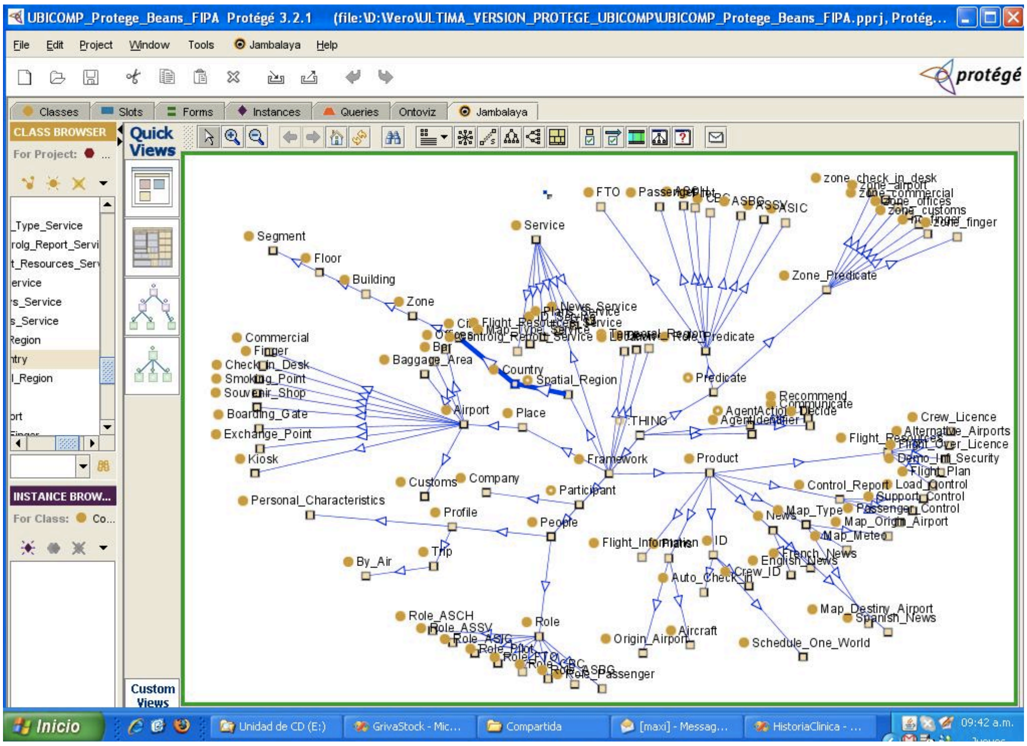


Figure 4.4: Ontology definition for Airport Domain

We have divided our Madrid-Barajas airport domain into six different zones that are not overlapped each other. Airport Zone (containing the rest of the zones), Customs Zone (customs), Commercial Zone (stores, cafeterias, restaurants), Offices Zone (airline offices), Check-In Desk Zone (check-in desks) and Finger Zone (finger). This is part of a previous work we have referenced before where we have used a centralized platform to define the behavior of our system. We firstly identified agent types as: Central, Provider and Client agents. Central agent represents the infrastructure that acquires location information from different sensors. Provider agents act on behalf of the different services and Client agents represents the users with a wireless device (see Figure 4.5) .

Later, we distinguished the roles of agents required in the system as: User Manager (UM), Location Information Manager (LIM), Agent Discover (AD), Service Provider (SP), Recommend Role (RR), Service Provider (SP) and Negotiation Role (NR). After that we defined three different ad-hoc protocols:

- **Receive-registry-profile**: Receive a request of registry of a user which has a shared part of the user profile.
- **Notify-agent**: Send an informative message to the closest provider in order to alert of the presence of a user.
- **Offer-service**: Send offers about services to users, that can be negotiate

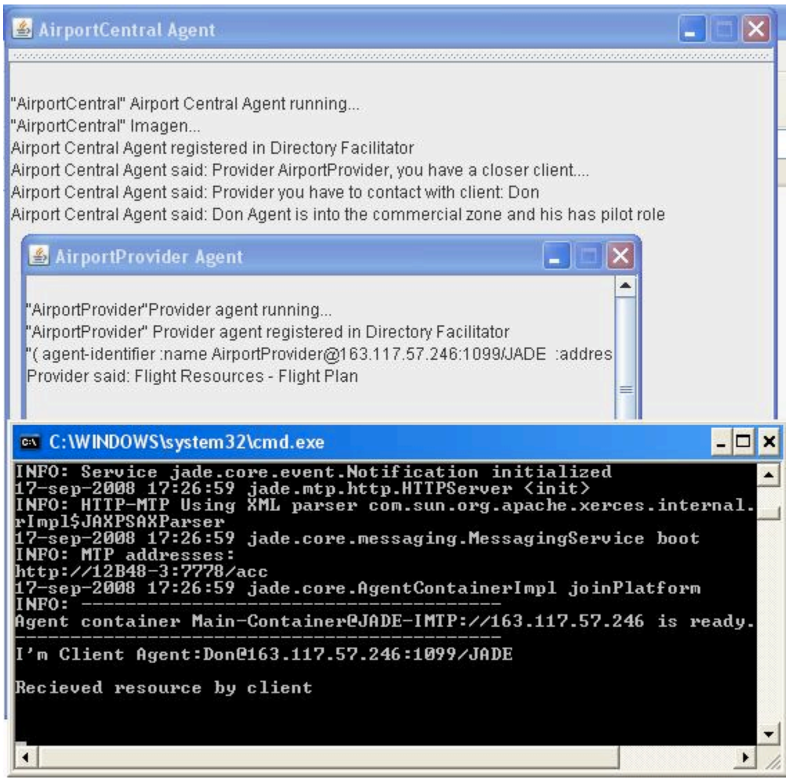


Figure 4.5: Agents communication in Airport Domain

4.1.3 System's features

Following there is a review of the basic functionalities we have implemented in our Airport Information System.

- Position awareness: The system can provide the users with information about the location of the different interest point in an airport. Moreover, the system may use the location information to provide the user with different services. The distribution of access points enables WiFi localization of the user within his home. The WifiPositionWidget queries the web server (for a certain WiFi MAC address), which successively calculates the user's position and sends back a structured data with this info.

- Context awareness: Services like Flight Plan, Alternative Airports document, Flight license and Crew license. Services are pushed to the pilot once in the airport zone.

- Schedule overview: Pilots can have insight of the scheduled activities of the crew. This is provided by interaction with the public Google Calendar API, where the crew has been appointed to the same flight.

- Weather forecast information: The pilot can be informed about the weather before leaving the airport. This is accomplished by acquiring data from the public API of the Google Weather service.

- News information: Users can be informed about the news before leaving the airport. It is accessed with the Newspaper URL.

4.1.4 Evaluation

Following the evaluation methodology for AmI systems and its three steps presented in the previous chapter, we define the purpose of the evaluation:

STEP 1 — Purpose

In the case airport domain (see Section 4.1) is a centralized system working to assist users of the airport area while they are carrying out their activities in different zones of the airport. We want to evaluate the system from the user perspective, so following (Abdualrazak et al., 2010) , the purposes of evaluation are:

- to measure the **application purpose** in terms of *usability*;
- to measure the **intelligence** in terms of context awareness, that means*quality of context*;

STEP 2 — Design

For the case of airport domain, and from the user centered parameters, we will measure the usability, considering how users perceive and utilize the airport information system. We want to measure if the functionalities satisfy users' expectations and full fill users' needs.

STEP 3 — Execution

We acquire the Effectiveness, Efficiency and Satisfaction parameters in order to measure usability (see Figure 4.6) by means of Myfeedback system[1]. According to the current position of the user, the services provided by the system, user can get connected offline to this system and score usability parameters and also he has the possibility to leave a comment.

4.2 Transportation domain: Railway information system (Griol et al., 2009b)

There is a high variety of applications in which conversational agents can be used. One of the most wide-spread is information retrieval. Some sample applications are tourist and travel information (Glass et al., 1995b; Os et al., 1999), weather forecast over the phone (Zue et al., 2000b), speech controlled telephone banking systems (Hardy et al., 2006; Melin et al., 2001b), conference help (Andreani et al., 2006; Bohus et al., 2007b), etc. They have

[1] http://giaa.inf.uc3m.es/airport/myfeedback/

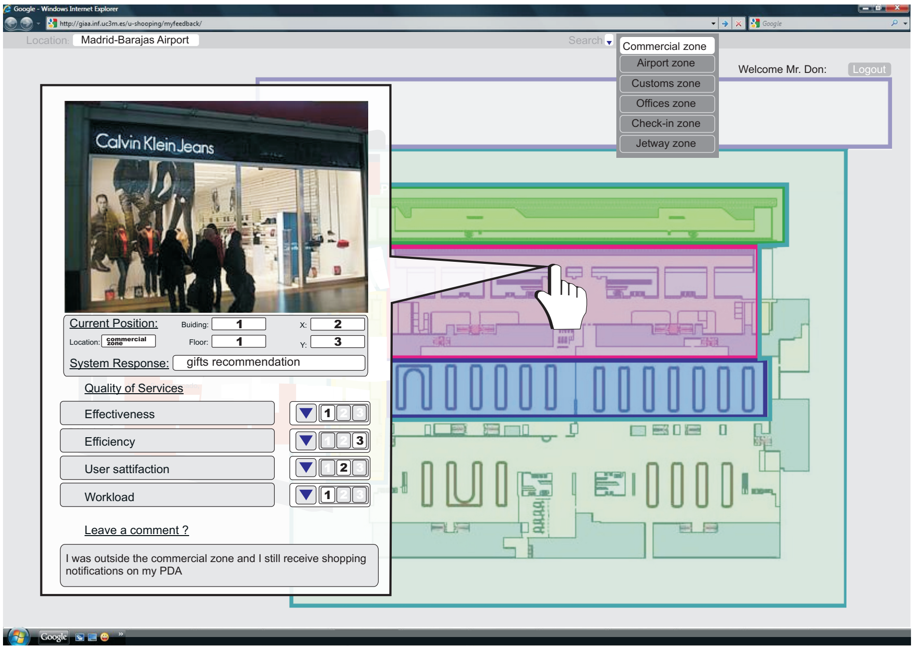


Figure 4.6: Acquiring usability variables.

also been used for education and training, particularly in improving phonetic and linguistic skills: assistance and guidance to F18 aircraft personnel during maintenance tasks (Bohus & Rudnicky, 2005), dialog applications for computer-aided speech therapy with different language pathologies (Vaquero et al., 2006).

In this section we present a railway information system that includes a speech-based interface. To successfully manage the interaction with the users, conversational agents usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialog management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules.

4.2.1 Overview of the railway/subway information system

The Railway/Subway Information System (RIS) was developed to provide with useful information to users of railway/subway stations: passengers and station staff (locomotive engineers, railroad/subway conductors and yardmasters, railroad/subway brake/signal/switch operators, supervisor). The system is focused on their key objectives – overall quality of information and rail services, personal security, transport capacity, train punctuality and a clean environment. The platform dynamically reacts to the context of each user. For example, the context-aware address book provides users with information on their closest available colleagues and the content is different depending on role and area. When they enter in the wireless network, the position of the client is evaluated and the systems initiates the negotiation of the set of applications the user can access depending on his physical position and preferences. To clarify the system's functionalities, following there are some of the scenarios where this Railway/Subway Information System can be found beneficial:

Scenario 1 — Railway/Subway station

Railway/Subway station equipped with location sensors to track the location of each unit in real-time. Based on the time needed to connect two locations with sensors, the system can also predict the speed of each unit. Examples of objects in this environment are tracks and stations. Interactors are trains, drivers and command centre officers. Sensors are used for identification purposes based on ID signals sent from the train. Other signals can be sent as well, e.g., emergency status. Actuators will be signals coordinating the flow of trains and messages that can be delivered to each unit in order to regulate their speed and the time they have to spend at a stop. Contexts of interest can be "delays" or "stopped train". One interaction rule can be " if line blocked ahead and there are intermediate stops describe the situation to passengers ".

Scenario 2 — Railway/Subway station

Patrick is a visual disabled user of the Railway Stations, he is used to take the same train at 10:00am, from Atocha to Torrelodones Station. He set his initial profile in the system (age: 29; gender: male; language: Spanish; kill level: high; pathologies: none. He also have preferences: Timetables; Talgo Train; Business Class; Atocha Station; 10:00am... On the other hand, Marcus is a railway engineer and his job is to take care of the maintenance of the railway, if a problem occurs during the operation, he is in charge to fix it. Patrick arrives early to the station and once he is connected to the Railway WLAN, the system switch to Speech Mode and starts a conversation with him in order to provide de correct service.

```
User name: Patrick
Location: Atocha Station
Date and Time: 2011-09-30, 9:00am
Device: PDAQ 00-18-41-32-0B-59
Objective: To know timetables and prices to Torrelodones station
```

4.2.2 Reasoning engine

The behaviour of the system is the following: In the first phase, the Aruba Positioning system discovers the user's position while he enters the Wi-Fi network in the Atocha Station. Later, positioning agent provides Aruba positioning information to the user agent. Once user agent knows its location sends it to the facilitator agent as well as the information regarding using a specific kind of service, in this case the user decides to ask for a conversational service. Following the set of phases, a conversational agent that provides the railway information and has been previously detected, asks the user agent about context information to be used during the interaction to provide the personalized service. Once this context information is received by the Context Manager included in the conversational agent, it loads the specific context profile characteristics. This information is then consulted by the rest of the modules in the conversational agent to personalize the provided service.

4.2.2.1 Ontology adequacy

The offering mapping is where offerings are mapped to zones and times. A mapping consists of a condition and an offering. The meaning of the mapping is that when the context of the user's device matches a condition, the client will be eligible for the offering During the context configuration procedure, we need to define every concept in our domain knowledge and represented in the ontology described in Figure 3.4. These concepts are:

- Location: XCoordinate int; YCoordinate int
- Place: Building ID int; Floor Level int
- Service: Name String
- Product: Name String; Characteristics: List of Feature
- Feature: Name String; Value String
- ProfileDescription: Name; Characteristics: List of String
- ProfileData: Name; Characteristics: List of Feature
- DialogLog: Log List of String

We have applied our context aware methodology to design an adaptive system that provides information in natural language about train services, schedules, and fares. The requirements for the task have been specified by taking into account the semantics defined for the DIHANA project (Griol et al., 2008). Users can ask for information about *Hour*, *Price*, *Train-Type*, *Trip-Time*, and *Services*. They also can provide task-independent information like *Affirmation*, *Negation*, and *Not-Understood*. The attributes needed by the system to answer to the different user queries are *Origin*, *Destination*, *Departure-Date*, *Arrival-Date*, *Ticket-Class*, *Departure-Hour*, *Arrival-Hour*, *Train-Type*, *Order-Number*, and *Services*. The system responses can be classified into the following categories: *Opening*, *Closing*, *Not-Understood*, *Waiting*, *New-Query*, *Acceptance*, *Rejection*, *Question*, *Confirmation*, and *Answer*. Once this package is received by the Context Manager, it loads the specific user profile. This information is then consulted by the rest of the modules in the dialogue system to personalize. For this example of scenario, the user profile contains the following information:

```
Name: José García
-------------------
Gender: Male | Age: 29 | Language: Spanish | Skill level: High | Pathologies: None...
-------------------
Sessions: 12 | Average
Turns: 5.7 | Average_Duration: 2.2 | Last_Session: 2009-04-29...
-------------------
Preferences: Timetables, Talgo Train, Business Class, Atocha Station, 10:00am...
-------------------
Current_Location: Atocha Station | Platform zone
```

In this case we have used an agent-based architecture including a Conversational Agent in order to provide a speech based interface. The first step is to execute the different

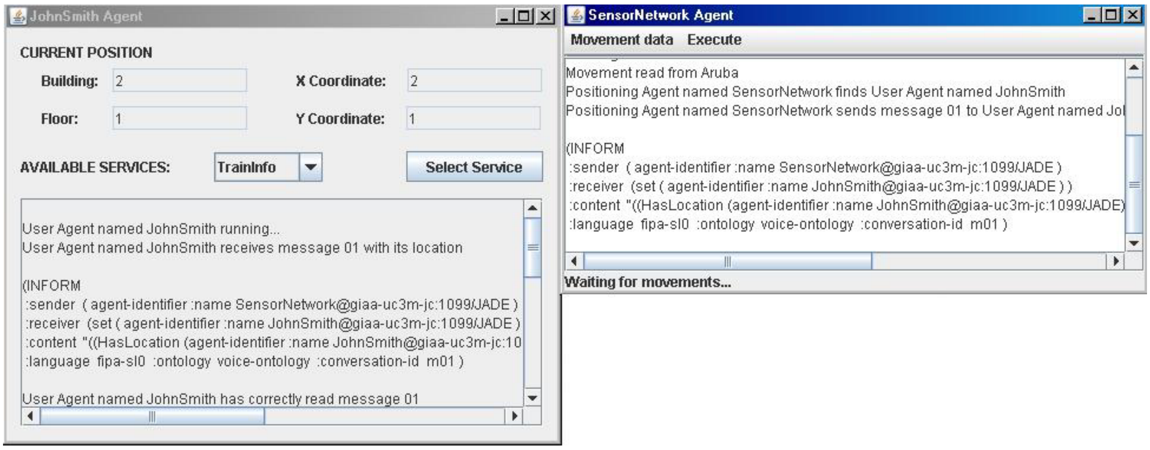


Figure 4.7: Positioning information sent from the Positioning Agent to the User Agent

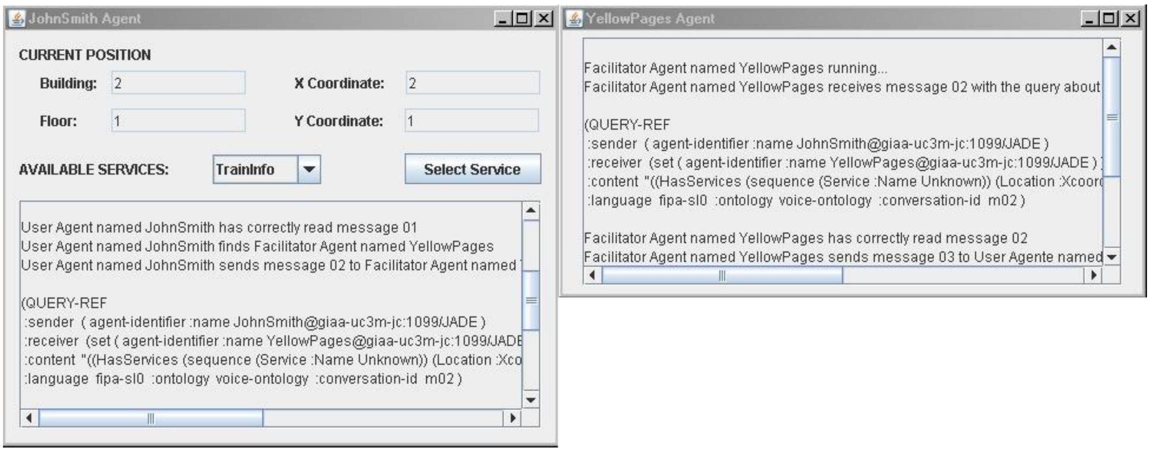


Figure 4.8: Query of the User Agent about the services that are available

agents using the JADE platform. Then the information about the position is sent from the Positioning Agent to the User Agent as shown in Figure 4.7. In the figure, the Positioning Agent is called Sensor Network and the name of the User Agent is John Smith. It can be observed the message that is sent by the Positioning Agent and received by the User Agent and how it establishes the new position values as shown at the top of Figure 4.7.

In the next phase, the User Agent John Smith looks for the Facilitator Agent (called "Yellow Pages") and ask it about the services that are available in the new location where the user is positioned. The message that is sent by the User Agent and received by the Facilitator Agent is shown in Figure 4.8.

Figure 4.9 shows how the Facilitator Agent "Yellow Pages" answers the User Agent John Smith by sending it the services that are available in the location of the user. This message is shown in the interfaces of the Facilitator and User Agents. The interpretation of this message by the User Agent makes possible that it shows the services that are available.

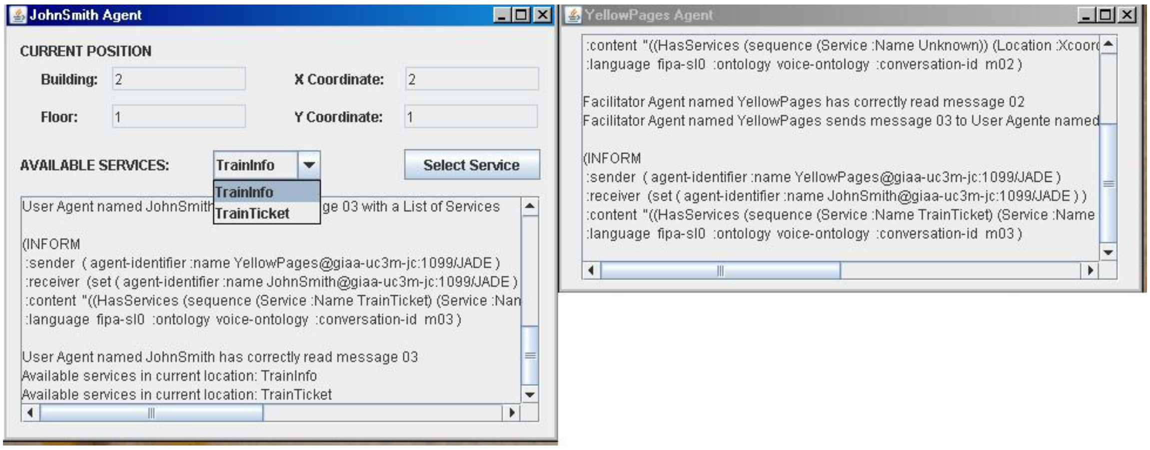


Figure 4.9: The facilitator Agent provides the services that are available

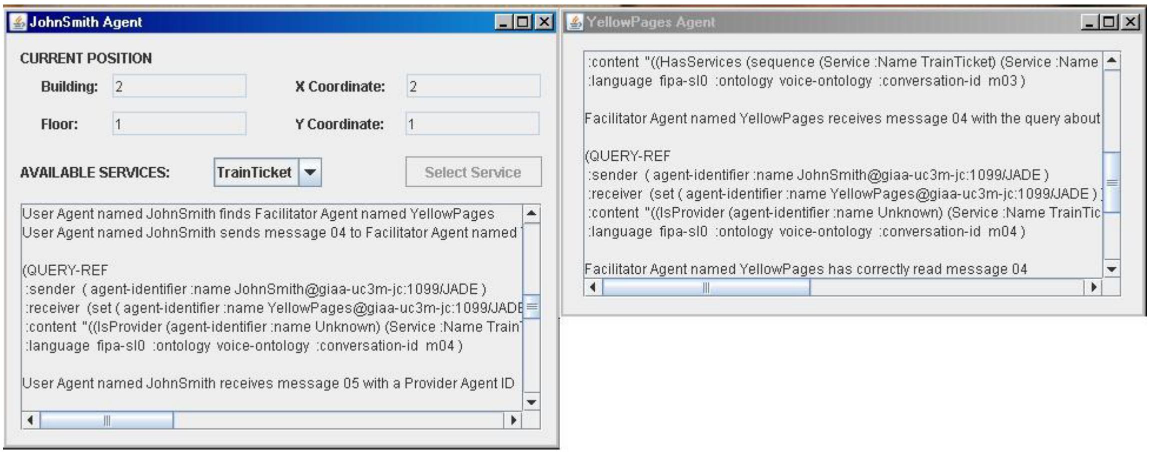


Figure 4.10: Selection of the required service by the User and facilitator Agents

Once this package is received by the Context Manager, it loads the specific user profile. This information is then consulted by the rest of the modules in the dialogue system to personalize .

The next phase consists of selecting the service, as shown in Figure 4.10. From the list of two services that are available in the current location (TrainTicket and TrainInfo), one of them has to be selected. This is the next phase in the communication process, in which the user asks the Facilitator Agent about the identifier of the Provider Agent for this service. The service TrainTicket is selected and the Facilitator Agent answers the user's query by communicating it the AID of the Provider Agent which provides the required service.

Facilitator Agent answers the user's query by communicating it the AID of the Provider Agent which provides the required service. Once the User Agent knows how to contact with the Provider Agent that supplies the service in which he is interested, it sends a query to be provided with service. Figure 4.11 shows this query to the Conversational Agent, called in the figure "Dialog System" Agent. Then, the Provider Agent asks the User Agent about the

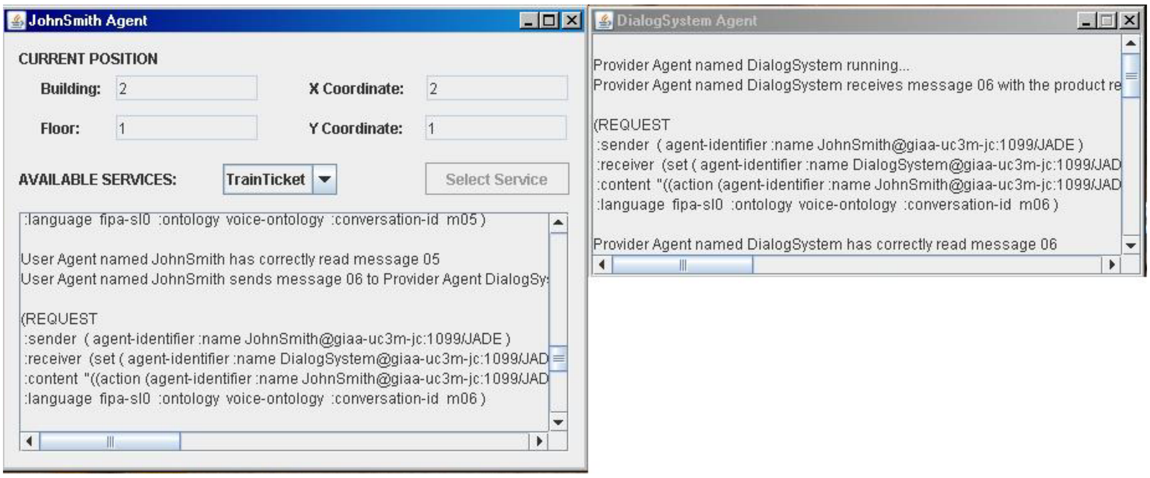


Figure 4.11: Query to the Conversational Agent that provides the service

context information that can be useful for the Conversational Agent to provide it with an adapted service.

User Agent decides which context information to send as a result of the previous query and transmits this information to the Provider Agent. Once the Conversational Agent has received the context information, it transmits this information to the Context Manager that is included in its architecture. The information defined for the context stored in the user agent can be classified into two different groups. The first group contains personal information (user's name, gender, age, current language, skill level when interacting with conversational agents, pathologies or speech disorders) and the second group stores information about user preferences using a specific service (statistic measures related to the previous dialogs that have been completed by the user, which are defined according to a specific domain-knowledge, i.e. in a railway information system to know that a specific user usually requires timetables information), or relative to specific values of attributes to be used in these queries, i.e. the preferences for travelling during a specific part of the day or by using a specific type of train).

The rest of modules of the Conversational Agent requires the Context Manager for the specific information that they require to adapt its behaviour. Once the interaction with the Conversational Agent has finished, this Agent communicates the result to the Log Analyzer Agent. When the result of the last dialogue has been interpreted, the agent adds this information to the set of previous dialogues that have been stored regarding the interactions of the User Agent John Smith with the Conversational Agent DialogueSystem. This way, this Provider Agent can provide the most adapted service taking into account the specific user'preferences detected in the previous dialogues. This process is automatically carried out, reducing the participation of the user in this process.

4.2.3 System's features

For this scenario, the user profile contains the following information (see Figure 4.12). Following there is a review of the basic functionalities we have implemented.

S1: Welcome to the railway information system. How can I help you?	*No prompts personalization.*
U1: I want to know timetables to Torrelodones.	
S2: From which station do you want to leave?	*There is not information about the current location of the user.*
U2: Atocha Station.	
S3: Please, tell me date.	*No date stored.*
U3: Today.	
S4: There are five trains leaving from Atocha Station to Torrelodones at 10:00am, 11:30am, 15:00pm and 19:00pm. Anything else?	*No information about the user preferences: timetables.*
U4: Tell me the price of the first one.	
S5: Please, tell me the ticket class.	*No information about the user preferences: ticket class.*
U5: Business class.	
S6: The price in business class is 2.5 euros. Anything else?	
U6: No, thank you.	
S7: Thank you for using the railway information system. Have a good trip!	

S1: Good morning Patrick, I consult you trains information leaving from Atocha Station. Do you want timetables information?	*Prompt personalization including the user's name. The user location is known and his preference about consulting timetables.*
U1: Yes, I want to know timetables to Torrelodones.	
S2: The following two trains leaves today at 10:00am and 10:27am from Atocha Station to Torrelodones. Anything else?	*The system uses the current hour and the date to personalize the timetables service. There is not any information about other possible queries performed by the user in previous dialogues.*
U2: Please, tell me the price of the first one.	
S3: The price in business class is 2.5 euros. Anything else?	*The system uses the ticket-class preferences included in the user profile.*
U3: No, thank you.	
S4: Thank you Patrick. Have a good trip!	*Prompt personalization including the user's name.*

Figure 4.12: An example of a dialogue

- Context awareness: Services like Timetables, Train list of prices are pushed to users once in the railway zone in a speech based interface.

- Sensor awareness: The system can provide the workers with information about the state of the railroad tracks in a speech based interface.

- Weathercast information: Passengers and workers from the station can be informed about any weather change in a speech based interface.

- News information: Users can be informed about the news services It is accessed with the Newspaper/Journal URL in a speech based interface.

4.2.4 Evaluation

Following the evaluation methodology for AmI systems and its three steps presented in the previous chapter, we define the purpose of the evaluation:

STEP 1– Purpose

In the case railway domain is a distributed system who implements also a dialogue system, working to assist users of the railway domain while they are carrying out their activities in different zones and with different preferences. Again, taking into account the taxonomy, the purposes of evaluation are:

- to measure the utility of the contextual information in this domain

STEP 2– Design

In orther to evaluate an information system we defined seven measures in order to prove the utility of the contextual information in this domain, so we make a comparison of the dialogues acquired using or not context information:

1. the percentage of successful dialogues
2. the average number of turns per dialogue
3. the percentage of different dialogues
4. the number of repetitions of the most seen dialogue
5. the number of turns of the most seen dialogue
6. the number of turns of the shortest dialogue
7. the number of turns of the longest dialogue

STEP 3– Execution

Using these measures, we tried to evaluate the success of our approach as well as its efficiency with regard to the different objectives specified in the scenarios. Table 4.13 shows the comparison of these measures. As it can be observed, the first advantage of our approach is regarding the number of dialogues that was necessary to simulate in order to obtain the 150 successful dialogues for each kind. While, only a 0.9% of successful dialogues is obtained in average for the S1 and S2 types without using context information, this percentage increases to 6.9% when the context manager is introduced. The second improvement is the reduction in the number of turns, as stated above in the dialogue examples. Using the context manager it is possible to obtain a reduction greater than 50% in the average number of turns per dialog. This reduction can also be observed in the number of turns of the longest, shortest and most seen dialogues. Finally, the number of different dialogues is lower using the context information due to the reduction in the number of turns, as can be observed in the number of repetitions of the most seen dialogue.

	Without Context Information		Using Context Information	
	Type S1	Type S2	Type S1	Type S2
Percentage of successful dialogues	1.6%	0.1%	11.4%	2.5
Average number of turns per dialogue	9.4	12.6	4.9	6.2
Percentage of different dialogues	92.9%	98.3%	71.9%	83.7%
Number of repetitions of the most seen dialogue	5	3	12	7
Number of turns of the most seen dialogue	7	9	5	7
Number of turns of the shortest dialogue	5	7	5	7
Number of turns of the longest dialogue	25	27	17	19

Figure 4.13: Results of the high-level dialogue features defined for the comparison of the two kinds of dialogues

4.3 U-health domain: Hospital (Sánchez-Pi & Molina, 2010)

The feasibility of using information and communications technologies (ICT), such as sensor networks, radio frequency identification (RFID) and Universal Mobile Telecommunications System (UMTS), has led to improve e-services and applications in the ambience of electronic healthcare (e-Health) (Wyatt & Sullivan, 2005). Patients and health centre workers(doctors, medical assistance, nursing, patient care services, health care services) aim to have an ubiquitous system capable of providing not only with updated information, but also with access to applications.

4.3.1 Overview of the U-health system

In the case of modern healthcare, it includes user mobility allowing people at risk or patients with proved health problems to continue their usual life at their homes and work places. Several initiatives, such as Mobihealth (Van Halteren et al., 2004), XMotion (Mentrup, 2004) and MyHeart (Philips Research, 2010) have investigated the feasibility and benefits of mobile healthcare services and applications. Furthermore, health care professionals also need to access and input medical or patient information from anywhere, at any time in their daily ward rounds (Haux, 2006; Johnson & Turley, 2006; Reuss et al., 2004). Hence, mobile healthcare systems can facilitate efficient and effective patient care information input and access at the point of patient care. Patients are assigned to a specific doctor once they arrive to the hospital. The Medical Record is available in Patients device, so when he arrives to the hospital, immediately the system sends a notification to Administration Office, who assigns a turn depending on his physical condition.

The platform dynamically reacts to the context of each user. For example, if there is a patient assigned to the context-aware address book provides users with information on their closest available colleagues and the content is different depending on role and area. When they enter in the wireless network, the position of the client is evaluated and the systems initiates the negotiation of the set of applications the user can access depending on his physical position and preferences. To clarify the system's functionalities, following there are some of the scenarios where this U-health Information System can be found beneficial:

Scenario 1 — Hospital

Hospital room, where a patient is monitored for health and security reasons. Objects in the environment are furniture, medical equipment, specific elements of the room like a toilet and a window. Interactors in this environment will be the patient, relatives and carers (e.g., nurses and doctors). Sensors can be movement sensors and wrist band detectors for identifying who is entering or leaving the room and who is approaching specific areas like a window or the toilet. Actuators can be microphones/speakers within the toilet to interact with the patient in an emergency. Contexts of interest can be " the patient has entered the toilet and has not returned after 20 minutes " or " frail patient left the room". Interaction rules can consider, for example, that "if patient is leaving the room and status indicates that this is not allowed for this particular patient then nurses should be notified".

Scenario 2 — Hospital

Margaret fell at home and had a lot of pain, her husband activates the system in his mobile device who queries the physical condition of the patient and provides with indications to get to the closest hospital. Once there, immediately the system sends a notification to the Administration Office, who assigns a turn depending on his physical condition. Doctor Joshua Brown was the one assigned who is an orthopedist. Dr. Brown diagnoses Margaret with a femur fracture, so she needs to go to surgery as soon as possible. She is assigned a bed in the orthopedic room, so she can be operated ASAP. Dr. Brown also have other patients in the same room, so while he's inside the hospital, he can monitor every patient in the room. He receives notifications with the condition state of every patient and he can update the treatment depending on the current vital signs of the patient or analysis results. Dr. Joshua indicates blood test to Margaret, results are pushed from the Laboratory to his PDA, so he can immediately indicates de surgery.

4.3.2 Reasoning engine

We use a multi-agent approach to implement the system. The behaviour of the system is the following: In the first phase, the Aruba Positioning system discovers the user's position(patient, doctors, nurses, medical assistant) while he enters the Wi-Fi network in the Majadahonda Hospital. Later, sensor agent provides user's positioning information to the user agent. Once user agent knows its location sends it to the facilitator agent as well as the information regarding using a specific kind of service. Following the set of phases, the facilitator agent communicates with provider agent, in this case Administration Agent which provides a turn to the Patient's agent to see a specialist doctor that has been also previously detected. Administration agent asks the patient agent about context information(medical condition, vital signs) to be used during the interaction to provide the personalized service. Once this context information is received by the Context Manager included in the provider agent, it loads the specific context profile characteristics. This information is then consulted to personalize the provided service.

The interaction between the different agents is described by the following sequence of phases:

- i) The Aruba Positioning system the patient's position while at WiFi home network and the Body Sensor Network provides the information regarding each sensor parameter;
- i) The sensor agent provides sensor information to the user agent;
- iii) Once the user agent knows its location, and its vital signs, sends it to the e-service facilitator agent. It also provides information regarding the type of user is using the system (patient/caregivers/patient's relatives);
- iv) The e-service facilitator agent sends the e-service provider agent the identification of the user agent that provides the kind of e-services required by the user agent;
- v) The user agent asks a specific e-service provider agent to provide it with the required e-service;
- vi) E-service provider agent asks the user agent about context information to be used during the interaction to provide the personalized e-service;
- vii) User agent provides the required context information to the e-service provider agent;
- viii) Interaction between the user agent and the e-service provider agent using the adaptation provided by the previous step;
- ix) Once the interaction and the provisioning of e-service tool place, evaluation agent ask the e-service facilitator agent about the type of user;
- x) Evaluation agent invokes Online/Offline evaluation system depending of the type of user;
- xi) Historical files are stored for analysis and feedback is provided into the system.

4.3.3 System's features

Following there is a review of the basic functionalities we have implemented in our U-health Information System.

- Sensor awareness: The system can provide the users with information about the location of the different zones in an hospital. Moreover, the system may also sense information in a Body Sensor Network and use this information to provide the user(patient/doctor/medical assistant) with different services. The distribution of access points enables WiFi localization of the user within the hospital.
- Context awareness: Services like Patient Medical Record, Medical treatment, Vital signs monitoring, Emergency call are pushed to users once in the hospital zone.
- Schedule overview: Doctors can have insight of the scheduled activities of the medical assistants of his room. This is provided by interaction with the public Google Calendar API, where the medical assistants has been assigned.
- Hospital statistics information: The doctor and personnel/medical assistant can be informed about any hospital statistics once inside the hospital.

- News information: Users can be informed about the news services It is accessed with the Newspaper/Journal URL.

4.3.4 Evaluation

Following the evaluation methodology for AmI systems and its three steps presented in the previous chapter, we define the purpose of the evaluation:

STEP 1 — Purpose

In the case u-health domain, it is a distributed system who implements, working to assist users of the hospital domain while they are carrying out their activities in different zones and with different preferences and roles. Again, taking into account the taxonomy, the purposes of evaluation in this case are:

- to measure the **application purpose** in terms of *usability*;
- to measure the **intelligence** in terms of context awareness, that means*quality of context*;

STEP 2 — Design

At the impact level, a possible metric would be the satisfaction rating of the user with a particular activity. Quality process has two distinct facets: technical quality and functional quality. Technical quality refers to the accuracy of medical diagnoses and procedures, and is generally comprehensible to the professional community, but not to patients (Bopp, 1990). Patients essentially perceive functional quality as the manner in which the service is delivered; while healthcare professional can be capable of making a technical quality evaluation.

There are several proposals regarding service quality measurement. Some of them are: SERVQUAL instrument proposed by Parasuraman (Parasuraman et al., 2002); SERVPERF (Gronroos, 1990) (Cronin Jr & Taylor, 1992); Yoo and Donthu (Yoo & Donthu, 2001) and Zhang and Prybutok (Zhang & Prybutok, 2005). Regarding this, for e-health environment we consider two groups of users: patients/caregivers/patient's relatives and health professionals. The first group will be able to make an online evaluation (OnE) of the system, for which we have defined some quality of context measurement, and the second one, an offline evaluation (OffE) with other quality of context measurement that evaluates, in this case, the technical quality of the system response.

STEP 3 — Execution

Main contributions regarding adaptation of user evaluation are:

- i) First, as the awareness of the system has been adapted for the e-health environment, the evaluation will be done based not only on the patients' location (as we did in (Sánchez-Pi & Molina, 2010b), but also on his vital signs: blood pressure (BP); pulse rate (PR); respiration rate (RR) and body temperature (BT). We based on the fact that the system is composed of a set of different sensors connected to a PDA that transmits, in a secure way, all the patient data (location and vital signs) to a central server in the hospital. The authorized doctors can access this medical information from their computers (inside the hospital or even outside) afterwards.

- ii) In the case of OnE, main contributions related to the adaptation of the service quality measurement to e-health are: Quality parameters measures the service quality gap between client expectations and perceptions of 5 quality attributes (on a five-point scale: strongly disagree = 1 to strongly agree = 5). Attributes are: easy of use; proceeding speed and effectiveness; reliability.

- iii) For OffE, we explore the e-service quality dimensions based on a review of the development of e-service quality scales and the SERVQUAL scale (Parasuraman et al., 2002). It proposes an 8-dimension scale but we will adapt this scale and add 2 more dimensions: system design (Appealing and well organized website; Consistent and standardized navigation; Well-organized appearance of user interface; Quickly downloading), reliability (Accurate delivery e-service; Complete order e-service; System being truthful about its diagnosis; The online e-service always correct; Keeping e-service promise; Accurate online e-service records; Website always available), fulfilment (Information on e-services available when need it; System runs smoothly in the transaction process; Accurate promises about delivery e-service when scheduled; Available to modify and/or defer the e-service process at any time without commitment), security (Protect the personal data of customers; Good reputation), responsiveness (Adequate contact information and performance; Prompt responses to customers; Timely responses to customers; Adequate response time), personalization, information and efficiency. Likerts's five point scale is used (strongly disagree = 1 to strongly agree = 5).

So, once the e-service is provided to the patient, the OnE evaluation system is invoked by the evaluation agent and patient/caregiver/patient relatives can make the evaluation of the e-services received filling the evaluation form. Doctor or health professional in charge of following the patients' file, can also evaluate the system behaviour as see in Figure 4.14. In this case Dr. makes an offline evaluation of the behaviour of the system during a week. Dr. suggests the system, in a similar case, to activate the DOCTOR EMERGENCY MEDICATION'S ALERT that will send a message to the doctor, so he can be notified immediately.

4.4 AAL domain: Intelligent home (Sánchez-Pi & Molina, 2009)

Ambient Assisted Living (AAL) born as an initiative from the European Union to emphasize the importance of addressing the needs of the ageing European population, which is growing every year as (United Nations, last access: Saturday, February 28, 2009; 12:01:46 AM.). The program intends to extend the time the elderly can live in their home environment by increasing the autonomy of people and assisting them in carrying out their daily activities. See

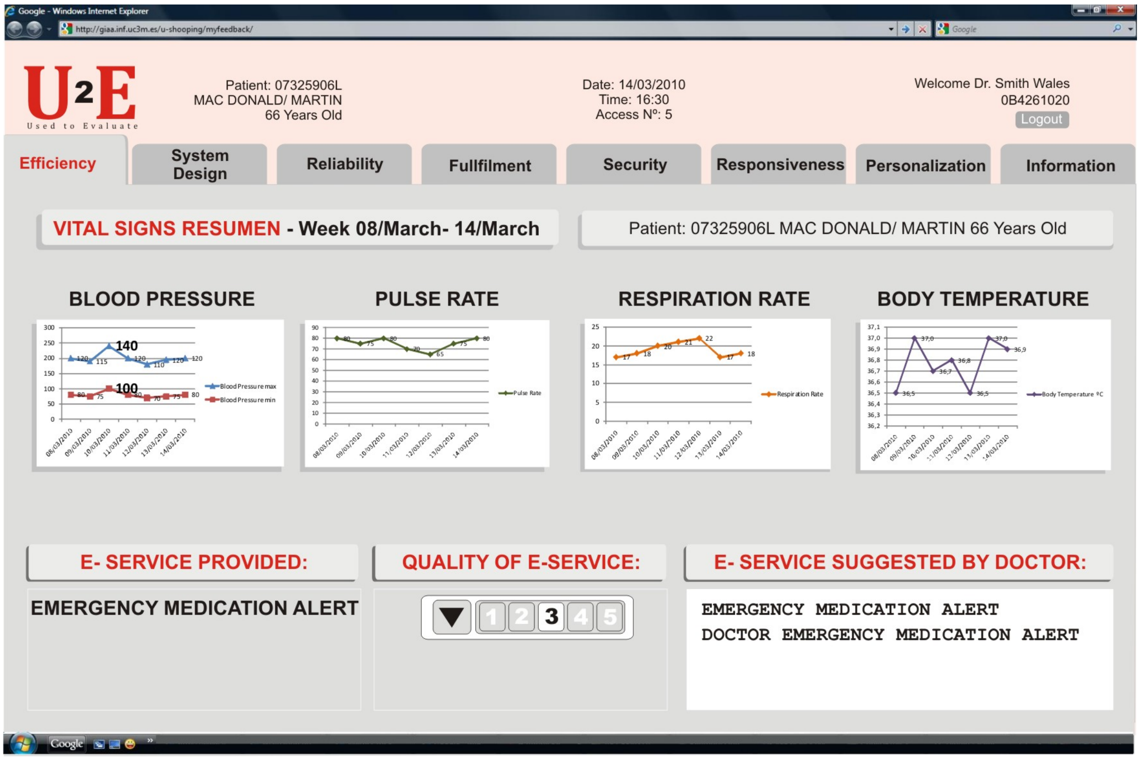


Figure 4.14: Online (left) and Offline (right) quality evaluation

Figure 4.15. Ambient Assisted Living (AAL) includes assistance to carry out daily activities, health and activity monitoring, enhancing safety and security, getting access to, medical and emergency systems. But home environments are challengeable as they're different to represent. There are no only elderly people in a home. There are also children and adults and there are multiple situations that can be represented. These include assistance to carry out their daily activities through smart object, like health or medication systems including wearable as well as context-aware services that facilitate activities like shopping or cooking. Ambient Intelligence has also been recognized as a promising approach to tackle the problems in the domain of Assisted Living (Emiliani & Stephanidis, 2005).

There have been several attempts of developing AAL systems. For example, (Kang et al., 2006) propose a wearable sensor system which measures the bio- functions of a person (heart rate, blood pressure, body temperature, body mass index, etc) to provide remote health monitoring and self health check that can be used at home. Korel and Kao (Korel & Kao, 2007) also monitor the vital signs and combine them with other context info such as environment temperature or person's condition, in order to detect alarming physical states and preventing health risks on time. (S-H. Baek, 2007) have designed an intelligent home care system based on a sensor platform to acquire data on heat and illumination. Taking into account the user's position, the home appliance control system manages the optimal performance of the devices at home (such as air conditioner, heater, lights, etc.). (H. Lee, 2006) implement a bundle of context-aware home services, ranging from doorbell answering services, seamless transfer of the TV image from one display device to another, reminders to turn of some devices while cooking or recipes outline on a display nearby. Healthcare and personal status monitoring applications are also common applications in AHCS; they usually imply the target user to wear sensors, and their main objective is to anticipate or detect health risks.

Furthermore, other systems aim at providing special care to a group of people with a certain disability. For example, (Helal et al., 2003) have developed a mobile patient care giver assistant deployed on a smart phone and responsible for catching the attention of people with Alzheimer's disease and notifying them about the next action they have to do. Medication prompting functionalities are also frequent in AHCS. Hardware to facilitate the medication consumption in the house has been developed, for example, by (Agarawala et al., 2004).

Moreover, several prototypes encompass the functionalities mentioned above: (Rentto et al., 2003), in the Wireless Wellness Monitor project, have developed a prototype of a smart home that integrates the context information from health monitoring devices and the information from the home appliances. (Becker et al., 2006) describe the amiCa project which supports monitoring of daily liquid and food intakes, location tracking and fall detection. The PAUL (Personal Assistant Unit for Living) system from University of (Floeck & Litz, 2007) collects signals from motion detectors, wall switches or body signals, and interprets them to assist the user in his daily life but also to monitor his health condition and to safeguard him. The data is interpreted using fuzzy logic, automata, pattern recognition and neural networks. It is a good example of the application of artificial intelligence to create proactive assistant environments. There are also several approaches with a distributed architecture like AMADE (Fraile et al., 2008) that integrates an alert management system as well as automated identification, location and movement control systems. static context referring to invariant features or dynamic context which is able to cope with information that changes. Static context is normally obtained directly from the user and the dynamic context indirectly from sensors.

4.4.1 Overview of the AAL system: Intelligent home/AHCS

The intelligent home scenario assist the elderly members of a family in their everyday life. Elders can specify personal activities they would like the house to automate (temperature control, light control, music control, etc.). For a grandfather sitting in a wheelchair with an RFID-tag, who usually take his medications between 10am and 11am, the following rule is discovered by the system:

Scenario 1 — Intelligent home/AHCS

The AmI specification may include the meaningful environment is the house, including the backyard and a portion of the front door as these areas also have sensors. When this environment is combined with e-heath services it is an Ambient Home Care Systems (AHCS). Objects are plants, furniture, and also three interactors: a person in the bedroom, a cat, and a floor cleaning robot in the living room. There are also movement sensors, pull cord switch, smoke detector, doorbell detector, pressure pad, plus switch sensors for taps, a cooker and a TV. In addition, there is a Body Sensor network installed to establish biometric data acquisition (gathered by a Bluetooth heart monitor) or movement estimation (provided from mobile accelerometers, in our case embedded in motes and in the heart monitor). A set of actuators, as the taps, cooker and TV also have the capacity to be turned on and off without human assistance. Contexts of interest can be "elder is left without caregiver presence ", "elder is still sleeping after 9AM". Interaction rules specified may consider that "if elder is in bed and is later than 9AM and he has not take his medication then caregiver or medical asistant should be notified".

```
User name: Patrick Smith (elder)
Location: bedroom
Date and Time: 2011-05-16, 9:0am
Device: PDAQ 00-37-11-88-1A-02
Objective: Taking medication at the proper time.
Turn on the TV and project the Medication Alert and
notify caregivers/medical assistant.
```

Scenario 2 — Intelligent home/AHCS

Contexts of interest in this case can be "blind elder is left without caregiver presence". Interaction rules specified may consider that "if elder has a routine doctor appointment alert then switch on PDA and speech mode and notify the alert".

```
Scenario 2: Rose Mary, Appointment Alert
User name: Mrs. Rose Mary (blind elder)
Location: kitchen
Date and Time: 2011-05-16, 10:30pm
Device: PDAQ 00-55-33-76-2S-45
Objective: Routine Doctor Appointment Alert. Turn on the PDA and the VoIP functionality will alert
through a voice message Mrs Rose Mary you have an appointment today with Dr. Princeton at 4pm".
```

These scenarios are for an intelligent home WiFi covered, and thought to assist not only the elder but also the members of the family. Our home environment is divided into several zones. See Figure: 4.15. The provisioning of the services occurs automatically in the Context Engine as the right context is found to each user: Role, Zone, Location, etc.

4.4.2 Reasoning engine

We use a multi-agent approach to implement the system. The behaviour of the system is the following: In the first phase, the Aruba Positioning system discovers the user's position(elders, caregivers, family members) while he enters the Wi-Fi network in the house. Later, sensor agent provides user's positioning information to the user agent. Once user agent knows its location sends it to the facilitator agent as well as the information regarding using a specific kind of service. Following the set of phases, the facilitator agent communicates with provider agent. And this one communicates with user agent about context information (medical condition, vital signs, failures detection) to be used during the interaction to provide the personalized service. Once this context information is received by the Context Manager included in the provider agent, it loads the specific context profile characteristics. This information is then consulted to personalize the provided service.

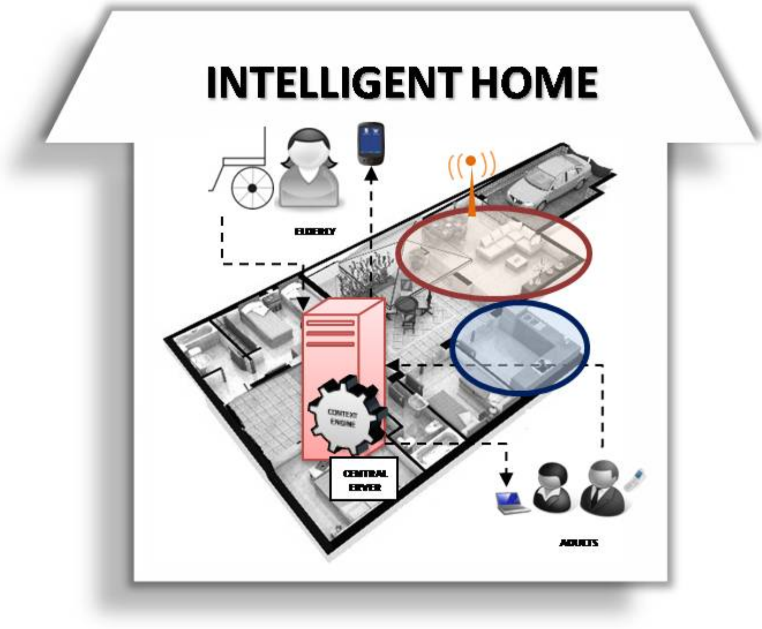


Figure 4.15: AAL scenario

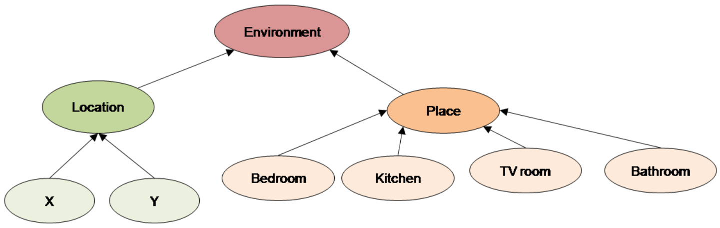


Figure 4.16: Environment Concept

We have adapted our ontology to cope with concepts of AAL domain. A user is an entity which interacts with the environment and other people. It is almost impossible to sense every entity in the environment because it is enormous. So, it is useless try to describe everything surrounding a user.

User mobility is a key concept in an AHCS domain, so we think location is an important concept in this part of the context specification requirements; see Figure 4.16.

- Location: XCoordinate int; YCoordinate int
- Place: Building ID int; Floor Level int
- Device: Device MAC; Sensor: Presence, Move, OpenDoor, Light, Temperature
- Service: Name String
- Product: Name String; Characteristics: List of Feature

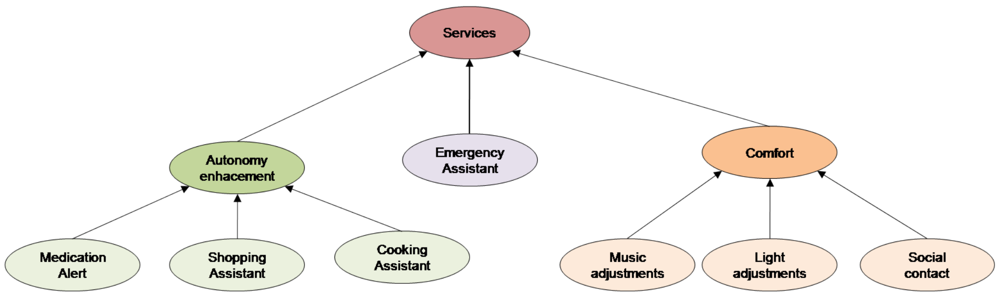


Figure 4.17: Services Concept.

- Feature: Name String; Value String
- ProfileDescription: Name; Characteristics: List of String
- ProfileData: Name; Characteristics: List of Feature

As context is only relevant if it influences the user and this is why the user takes an important place in AmI. This concept will have static facts like: gender, name and age and will also two important concepts to be taken into account: the role the user can have into the system and its preferences which contain the dynamic information of the user. Both concepts will determine which service should be available to which user as well as some other environment requirements. Role concept can be: elderly and it will determine a set of common characteristic. And the user's preference will be subject to the current situation, that's why it is more or less dynamic. Is in this concept here users can specify personal activities they would like the house to automate (temperature control, light control, music control, etc.) or the services he would like to receive.

Offerings contain several categories of services with similar characteristics. These services might be adapted to the user's preferences and to the environmental conditions. Categories in the system can be structured into comfort category including light and music adjustments, social contacts service and a special service designed just for children where music, images, light and sound are used to transform the children bedroom in a special space. Another category is the autonomy enhancement including services like: medication, shopping and cooking mainly addressed to elderly people. And finally the emergency assistant category designed for the assistance, prediction and prevention of any emergency occurred related to any member of the family. See Figure 4.17

4.4.3 System's features

- **Position awareness:** The system can provide the caregiver with information about the location of the patient/elderly and other people (assistants or visitors) within his home. Moreover, the system may use the location information to provide the user with different services.
- **Schedule overview:** Caregivers can have insight of the scheduled medication intake events of the patients.

- **Environmental parameters:** The caregiver's monitoring tool features information about the state of the user's home environment providing information about parameters such as: light, temperature and humidity.
- **Weather forecast information:** The patient can be informed about the weather before leaving the house.
- **Context-aware medication intake reminder:** Users at home should be able to get notifications for each medication intake scheduled in the calendar. The reminder issued depends on the patient's state and location. Users at home also receive a reminder for medication intake when overlapping events scheduled on different locations are found.
- **Biomedical functions:** The caregiver's application monitors the ECG, heart rate, blood pressure, oxygen saturation and sweat volume/rate, and combine them with other context info such as environment temperature or person's condition, in order to detect any alarming physical states and prevent health risks on time.

4.4.4 Evaluation

The evaluation of AAL domain from the users' perspective is very similar to the one used in the hospital domain.

STEP 1 — Purpose

In the case AAL domain, it is a distributed system who implements, working to assist users of the AAL domain while they are carrying out their activities in different zones and with different preferences and roles. Again, taking into account the taxonomy, the purposes of evaluation in this case are:

- to measure the **application purpose** in terms of *usability*;
- to measure the **intelligence** in terms of context awareness, that means*quality of context*;

STEP 2 — Design

For the case of AAL domain, and from the user centered parameters, we will measure the usability, considering how users perceive and utilize the airport information system. We want to measure if the functionalities satisfy users' expectations and fullfill users' needs.

STEP 3 — Execution

We acquire the Effectiveness, Efficiency and Satisfaction parameters in order to measure usability by means of Myfeedback system (http://giaa.inf.uc3m.es/aal/myfeedback/). According to the current position of the user, the services provided by the system, user can get connected offline to this system and score usability parameters and also he has the possibility to leave a comment.

4.5 Education domain: University (Sánchez-Pi et al., 2010)

AmI scenarios for a specific domain, the university that represents variable environment where many people interact with numerous systems and devices, were proposed in (Mikulecky & Olsevicova, 2005). Students, teachers, managers, librarians, administrative staff and others need to access information and knowledge from different sources, according their individual needs and in different situations. These information and knowledge-intensive activities are limited by numerous constraints in terms of location, time and availability. University is a good place for demonstration of general AmI scenarios reusing UbiComp principles, push and pull technologies and new types of devices.

4.5.1 Overview of the university information system

Scenario 1 — University

University where students are monitored on balancing their learning experience. The objects within a classroom or play ground are tables and other available elements. The interactors are students and teachers. The sensors will identify who is using what scientific kit and that in turn will allow monitoring of how long students are involved with a particular experiment. Actuators can be recommendations delivered to wristwatch-like personalized displays. Contexts of interest can be "student has been with a single experimentation kit for too long" or "student has not engaged in active experimentation". The first context will trigger a rule "if student has been interacting with one single kit for more than 20 minutes advise the student to try the next experiment available" whilst the second one can send a message to a tutor, such as "if student has not engaged for more than 5 minutes with an experiment then tutor has to encourage and guide the student".

Scenario 2 — University

University where students use a dialogue information system. Miss. Angie Lion, is a Computer Science student. She has AI exam next Wednesday and she wants to locate Prof. Peter Smith in order to appoint a tutoring with him. The first context will trigger a rule " if student is arriving late to the tutoring then will send a message to the tutor to let him know"; on the other hand show an interactive plan of the location of Prof. Smith office.

```
User name: Angie Lion
Location: Campus University, Main Building,
Side A, First Floor
Date and Time: 2009-11-03, 9:00am
Device: PDAQ 00-1C-41-32-0A-59
Objective: To know the location of the office of the professor Peter Smith.
```

4.5.2 Reasoning engine

We have applied our context aware architecture to design and evaluate an adaptive system that provides information in an academic domain (Griol et al., 2009a). This information can be classified in four main groups: subjects, professors, doctoral studies and registration. The system must have gathered some data by asking the user about the name of the subjects, degrees, groups names, professors, groups, semesters, name of a doctoral program, name of a course, or name of a registration deadline. The way in which the user is queried for this information follows in most cases a system-directed initiative. A set of four different scenarios has been used to evaluate our proposal for this task, taking into account the four different queries that a user can perform to the system.

As summarized in the description of our architecture, a positioning agent is used to determine user's position while he enters the WiFi network in the University campus. Then, the conversational agent that provides the academic information (and has been previously detected) asks the user agent about information for his identification to provide the personalized service. Once this identification is received by the context manager included in the conversational agent, it loads the specific context profile characteristics. This information is then consulted by the rest of the modules in the conversational agent to personalize the provided service. For this example of scenario, this profile contains the following information:

```
Name: Angie Lion
-------------------
Gender: Female | Age: 21 | Language: Spanish | Skill level:
High | Pathologies: None...
-------------------
Preferences: Computer Science, Tutoring Information...
-------------------
Current_Location: Main Building, Side A, First Floor
```

4.5.3 System's features

Figure 4.18 shows a dialog for this task acquired without incorporating the context manager, and a dialog for the same scenario acquired using our approach. It can be observed how the different system answers can be personalized by taking into account the information provided by the context manager. The results showed that the system shows a tendency of answering with higher agility and more natural than without including the context manager.

4.5.4 Evaluation

The evaluation of University domain is very similar to the one of the railway station because they both use a dialogue system to interact with the user, so our goal will be evaluate the utility of the contextual information for the dialoue

STEP 1 — Purpose

In the case railway domain is a distributed system who implements also a dialogue system, working to assist users of the railway domain while they are carrying out their activities in different zones and with different preferences. Again, taking into account the taxonomy, the purposes of evaluation are:

- to measure the utility of the contextual information in this domain

S1: Welcome to the university information system. How can I help you?	*No prompts personalization.*
U1: I want to know information about the tutoring sessions of Professor Peter Smith.	
S2: Can you tell me the name of the subject?	*There is not information about the specific subject.*
U2: Language Processors.	
S3: The tutoring sessions of Professor Smith for Language Processors are Monday from 14:00 to 16:00 and Wednesday from 11:00 to 14:00. Anything else?	
U3: Can you tell me the office number?	
S4: The office number is 2.1.B.06 in the Main Building. Anything else?	*No information about the current position of the user.*
U4: No, thank you.	
S5: Thank you for using the university information system. Have a nice day!	
S1: Good evening Patricia, do you want to know tutoring information?	*Prompt personalization including the user's name. The system knows the preference of the user about consulting tutoring sessions.*
U1: Yes, I want to know the tutoring sessions of Professor Smith.	
S2: The tutoring sessions of Professor Smith for Language Processors are Monday from 14:00 to 16:00 and Wednesday from 11:00 to 14:00. Anything else?	*There is information about the specific subject.*
U2: Can you tell me the office number?	
S3: The office number is 2.1.B.06. You just have to walk along the Side A corridor and turn left. The office is the second door on the right. Anything else?	*The system uses current location of the user to personalize the answer.*
U3: No, thank you.	
S4: Thank you Patricia. Have a nice day!	*Prompt personalization including the user's name.*

Figure 4.18: An example of a dialogue for the academic domain without adding the context manager (above) and including this module in our architecture (below)

STEP 2 — Design

In order to evaluate an information system we defined seven measures in order to prove the utility of the contextual information in this domain, so we make a comparison of the dialogues acquired using or not context information:

1. the percentage of successful dialogues
2. the average number of turns per dialogue
3. the percentage of different dialogues
4. the number of repetitions of the most seen dialogue
5. the number of turns of the most seen dialogue
6. the number of turns of the shortest dialogue
7. the number of turns of the longest dialogue

STEP 3 — Execution

Metrics are executed as in Railway scenario.

4.6 Commerce domain: U-Commerce (Sánchez-Pi & Molina, 2010b)

The evolution of e-commerce into m-commerce is leading to a world of ubiquitous commerce (u-commerce). The trend is for ubiquitous universal access via multipurpose terminals over sensor networks or RFID technology, which have the ability to identify, track and trace objects automatically, permitting the delivery of content over any network (Asif & Mandviwalla, 2005); (Ohkubo et al., 2005). Context is a central key in ubiquitous commerce (Coutaz et al., 2005). This content delivery can be adapted to the unique context of the person, the time, the place and the network and can act in unison in order to support smarter and more intelligent delivery (Lyytinen & Yoo, 2002), (Russell et al., 2005). This trend was named U-commerce and is facilitated by the emergence of four U-forces: ubiquity, universality, uniqueness and unity. Watson defines U-commerce as the use of ubiquitous networks to support personalized and uninterrupted communications and transactions between an organization and its various stakeholders to provide a level of value over, above and beyond traditional commerce (Watson et al., 2002).

The trend toward u-commerce does not represent simply a change in the way customers access and use information. In the end it will have a deep impact on the way customers use services, enabling new classes of services that only make sense by virtue of being embedded in the environment.

4.6.1 Overview of the u-commerce system

In this section, we illustrate our multi-agent context-aware system for a u-commerce scenario.

Scenario 1 — U-commerce

A young customer "John" goes to "Parquesur" shopping mall because he needs to buy some new clothes. Every store into the shopping mall hosts a vendor agent that is used to interact with shopper agents in order to suggest and offer the best product to the clients using the broker agent. Once in the shopping mall, they immediately connect their wireless PDAs to Parque Sur wireless network. Once connected, the shopper agent (used to provide a way to interact with the client (through a GUI)) is installed into the PDAs and begins the registering process and the provisioning of services.

```
User name: John Mayer
Location: Parquesur Shopping Center,
Side A, First Floor
Date and Time: 2011-11-03, 12:00am
Device: PDAQ 00-18-41-32-0B-59
Objective: To buy some new clothes, find new offers
```

4.6.2 Reasoning engine

There are several approaches to intelligent agents in the field of U-commerce; they are distinguished by their inherent characteristics: autonomy, reactivity, proactivity and mobility. Thus, intelligent agents represent a logical choice for using intelligence in the u-commerce applications. The challenge, therefore comes from the idea of offering an intuitive and conceptually simple model in which customers can achieve the objective of receiving the adequate service as they are paying for it. As the next generation business model, u-commerce immediately triggered a lot of attention among enterprises.

As in previous works (Fuentes et al., 2006a), (Fuentes et al., 2006b) we have used an ontology to describe contextual information including location, profile, preferences, devices, and network etc. In this case study, however, we have conceived context aware system as an interactive model between customers (shoppers) and vendors. Thus, we need to address the context description of shoppers and vendors. We have, therefore, developed two types of context ontology: shopper ontology and vendor ontology following (Yang, 2006) but we instantiate ours.

We have applied Protégé (Protégé Project. http://protege.stanford.edu/) to build the shopper ontology and vendor ontology; the major difference between both ontologies lies in their profiles. The shopper ontology contains the shopper's profiles, such as personal profile, accessibility and preferences, calendar profile, social profile, and location profile; the vendor ontology contains service profiles, such as input, output, pre-condition, and effect of service execution. As we are interested in the customer's evaluation, we focused on the shopper's ontology concepts. The shopper's ontology includes the following principal concepts:

- Location: XCoordinate int; YCoordinate int
- Place: Building ID int; Floor Level int
- Device: Device MAC; Sensor: Presence, Move, OpenDoor, Light, Temperature
- Service: Name String
- Product: Name String; Characteristics: List of Feature
- Feature: Name String; Value String
- ProfileDescription: Name; Characteristics: List of String
- ProfileData: Name; Characteristics: List of Feature
- Calendar profile: Name; Characteristics: List of Feature (owner, event, time, attendee, location)
- Social profile: Name; Characteristics: List of Feature (owner, Name, ID, Privacy, Collaborator, Proficiency, Trust)

In addition to profiles, which include personal, calendar and social profile, both the shopper ontology and vendor ontology contain a surrounding context, such as quality of shopping services profile. Quality of shopping services profiles contain constraints that can be described by response time, reliability, availability, and cost. They can have values like: (1) Correct, (2) Different order or (3) Incorrect. These will later help us to evaluate whether or not the system delivers the adequate services.

First, we present the redesign of a multi-agent system developed in a previous paper for a specific AmI scenario (Fuentes et al., 2006b; Sánchez-Pi et al., 2008a). The redesign includes new features to support u-commerce services and applications where customers will become "shoppers" and where three new categories of agents appear: "vendor", "broker" and "positioning". For a prototype version of the Shopping Recommender System, we define three agent types as explained in last section: Shopper agents, Vendor agents and a Broker agent. As in any distribution channel, different Vendors may collaborate/compete for the provider role. We have used the BDI agent model to specify agent architecture able to deal with environment uncertainty and with graded mental attitudes. Belief degrees represent to what extent the agent believes a formula is true. Degrees of positive or negative desire allow the agent to set different levels of preference or rejection respectively. Intention degrees also give a preference measure but, in this case, modeling the cost/benefit trade off of reaching an agent's goal.

The main contributions related to the adaptation of the MAS architecture to u-commerce are:

- in the new proposal, we remove the presence of a central agent that was in charge of centralizing the information exchanged between agents;
- user agent becomes a shopper agent in the new proposal and its main goals include negotiation with vendor agents, the recommendation of services to other shopper agents, trust in other agents, and management and improvement of their internal profile to receive better services according to that profile;
- provider agents become vendor agents in the new proposal and their functionalities are closely related to negotiating with shopper agents, since they can reach an agreement with shoppers and communicate and provide the most suitable services to them, according, of course, to the shopper's preferences and profile;
- there is a new figure in our architecture: the broker agent, whose main task is to act as a proxy between the shopper agent and the vendor agent, passing on a pseudonym in place of the shopper's real identity.

This agent is also in charge of matching shopper's preferences and vendor's services. Selected personal information, such as the current location of the customer, can be queried by the vendor from the broker agent, using a pseudonym. A persistent pseudonym value allows the broker agent to maintain personal information for shoppers, without compromising their privacy throughout interactions with different services.

The interaction between the different agents is described by the following sequence of phases:

1. The Aruba Positioning system discovers the customer's position when he enters the Wifi network

2. The positioning agent provides Aruba positioning information to the shopper agent;
3. Once the shopper agent knows its location, it sends it to the broker agent. It also provides information regarding the use of a specific kind of e-service;
4. The broker agent sends the shopper agent the identification of the vendor agent that provides the kind of e-services required by the shopper agent;
5. The shopper agent asks a specific vendor agent to provide it with the required e-service;
6. Vendor agent asks the shopper agent about context information to be used during the interaction to provide the personalized e-service;
7. Shopper agent provides the required context information to the vendor agent;
8. Interaction between the shopper agent and the vendor agent using the adaptation provided by the previous step.

4.6.3 System's features

Following there is a review of the basic functionalities we have implemented in our U-Commerce Information System.

- Sensor awareness: The system can provide the users with information about the location of the different zones of the shopping mall.
- Context awareness: Services like Product Offering, Negotiation are pushed to users once in the hospital zone.
- Shooping mall statistics information
- News information: Users can be informed about the news services It is accessed with the Newspaper/Journal URL.

4.6.4 Evaluation

STEP 1 — Purpose

In the case u-commerce domain, it is a distributed system who implements, working to assist users of the ubiquitous commerce domain while they are carrying out their activities in different zones and with different preferences and roles. Again, taking into account the taxonomy, the purposes of evaluation in this case are:

- to measure the **application purpose** in terms of *usability*;
- to measure the **intelligence** in terms of context awareness, that means*quality of context*;

STEP 2 — Design

At the impact level, a possible metric would be the satisfaction rating of the user with a particular activity.

STEP 3 — Execution

Main contributions regarding adaptation of user evaluation are:

- i) First, as the awareness of the system has been adapted for the e-health environment, the evaluation will be done based not only on the patients' location (as we did in (Sánchez-Pi & Molina, 2010b), but also on his vital signs: blood pressure (BP); pulse rate (PR); respiration rate (RR) and body temperature (BT). We based on the fact that the system is composed of a set of different sensors connected to a PDA that transmits, in a secure way, all the patient data (location and vital signs) to a central server in the hospital. The authorized doctors can access this medical information from their computers (inside the hospital or even outside) afterwards.
- ii) In the case of OnE, main contributions related to the adaptation of the service quality measurement to e-health are: Quality parameters measures the service quality gap between client expectations and perceptions of 5 quality attributes (on a five-point scale: strongly disagree = 1 to strongly agree = 5). Attributes are: easy of use; proceeding speed and effectiveness; reliability.

It is also important to have an approach to the Schmidt proposal, but dynamically introduce the online customer's evaluation. It allows the customer to have dynamic access to specific context concepts in order to modify them. It also offers the possibility of proposing a modification to the reasoning algorithms and, of course, to the inference rules executed. As we can make inferences about what a customer needs, the evaluation must also take into account the correctness of our reasoning. If we draw incorrect conclusions, the customer will likely receive incorrect information. For instance, if the basic task the customer is carrying out and there is some knowledge of the information needed for that task. One possibility would be to record the situational information the customer gives about doing this particular task and take note of what actions the customer is performing at a particular time. This could be stored as part of the quality of shopping/vendor services concept in our ontology, as an historical file. It can later be analyse through the user's opinion, which is crucial to these kinds of systems.

With regard to quantitative evaluation, customers may be measured either quantitatively, typically by defining a utility function and mapping the "satisfaction rate" to numeric value, or qualitatively. The value of quality of shopping/vendor services corresponds to the user feedback, and can have values like: (1) Correct, (2) Different order or (3) Incorrect, for every attribute.

This opinion is given after he receives the ranking of the products the system recommends. We want to know whether the broker agent is a personal agent satisfying, in some degree, a set of different users. As the process of information classification is generally a complex and personal task, and may differ among individuals, we can measure the average system behaviour over a population. We can then compare the inferences made by looking at the situational data and the actual customer request to draw some conclusions.

The preferences and restrictions introduced by customers as input to the system, together with the system results and the user feedback, constitute our N-cases set. Each case in the dataset will be composed of:

- User Input: a user ID and his graded preferences and restrictions.
- Agent's Result: the system returns a ranking of a maximum of ten products and offers.
- User Feedback: as explained previously, after analyzing the information of the recommended plans, the user provides feedback by evaluating the results as: (1) Correct, (2) Different order or (3) Incorrect.

In order to give a general measure of the broker agent's results over the satisfactory cases, we evaluate how close the broker agent's ranking is to the customer's own ranking. For this, we chose the Manhattan distance between the position of the first three products selected by the user and their position in the system ranking. In (Schoenharl at al., 2008) there is a study evaluating the applicability of several different measures to the validation of Agent-Based Modelling simulations, which is also applied in (Bruijns, 2009). From a computational perspective, Manhattan distance is significantly less costly to calculate than Euclidean distance, as it does not require taking a square root. So, this distance was adopted because it has proved to be appropriate for capturing positional differences. If we assume the quality of shopping agent (4.1), that is the user's feedback, is:

$$Q_i = (P_i 1; P_i 2; P_i 3) \tag{4.1}$$

and the broker agent ranking for this consult is (4.2):

$$R_i = (R_1; R_2; ..., R_n) \tag{4.2}$$

Then, if (4.3)

$$P_i 1 = R_j, P_i 2 = R_k; P_i 3 = R_n \tag{4.3}$$

the distance between the customer's and the system rankings is defined by (4.4):

$$d(\overline{p}, \overline{q}) = \sum_{i=1}^{n} |p_i - q_i| \tag{4.4}$$

In reference to the quality of information, user feedback can be expressed by writing a comment about the satisfaction or dissatisfaction experienced with regard to a particular aspect, or with a suggestion to improve the feedback. Both quantitative (e.g., effectiveness, efficiency, user satisfaction, and workload) and qualitative (e.g., user comments) data were obtained from the user.

So our system could then: i) discriminate between contextual information; ii) allow customers to rewrite the concept quality of shopping service described in the ontology (which will be mapped to obtained a quantitative evaluation); iii) allow customers to make personalized annotations and leave them in the current location (qualitative); iv) introduce a new concept in the ontology, such as the situational information concept described above.

4.6.4.1 Example of evaluation in u-commerce

A young customer "John" goes to "Parquesur" shopping mall because he needs to buy some new clothes. Every store into the shopping mall hosts a vendor agent that is used to interact with shopper agents in order to suggest and offer the best product to the clients using the broker agent. Once in the shopping mall, they immediately connect their wireless PDAs to Parque Sur wireless network. Once connected, the shopper agent (used to provide a way to interact with the client (through a GUI)) is installed into the PDAs and begins the registering process. The Broker agent then assigns a pseudonym to the shopper agent to preserve its identity. Processing of location information provided by the corresponding agent.

In the case of John, who is a new client, the broker agent emits some required questions to register an initial minimal by means of a preference questionnaire. Other additional data can be included in the profile, but will be considered optional, such as: address, phone, email, and ID number. John, however, decided not to respond, although some of this would be considered private information, and, therefore, would only be used internally, by the shopper agent to provide a customized filter of the services offered by vendor agents. Next, the broker agent asks the shopper agent these and other questions related to the preferences connected with the current shopping activity, such as: Gender, Age, Clothes, Shoes, and Jewelry. Once Receive-registry-profile protocol has concluded, the broker agent evaluates its position and computes the geographical proximity to the location of different vendors' agents offering the shopper's agent the possibility of downloading a discount coupon for using our systems. Calvin Klein's vendor agent is the closest to John's location, so the broker agent notifies it about the presence of John's shopper agent with the 'Notify agent' protocol. When Calvin Klein's vendor agent has been notified, it will provide its particular offers and customized suggestions according to the context exchanged with the broker agent. After that, a negotiation included in the offer-service protocol takes place in order to reach a possible agreement about services. For instance, Calvin Klein's vendor agent makes a match between John's preferences and the concepts stored in our ontology and immediately informs the shopper agent about two types of jeans, model A, model B.

John is surprised by the vendor agent's suggestions, because both of them are for women's jeans, so he uses the customer evaluation interface to make some annotations in the parameters of "quality of shopping service" icon present in the GUI to introduce a value in the "reliability" field, which remains as a historic field. The Vendor agent then informs the broker agent about the misunderstanding, and then broker agent asks the shopper agent to fill in the optional fields of the preferences questionnaire. It is then that the gender field is filled out and the system can suggest the proper information to the shopper agent.

Later, John arrives home, sits in front of the computer, opens the browser and connects to www.giaa.inf.uc3m.es/u-shopping/myfeedback. He logs-in and selects from the category of "shopping malls", the one he has visited recently. He clicks on the specific store he wants to comment upon and evaluates the contextual information he received by means of an e-service in a certain place. A picture of the front of the store appears, confirming its location and

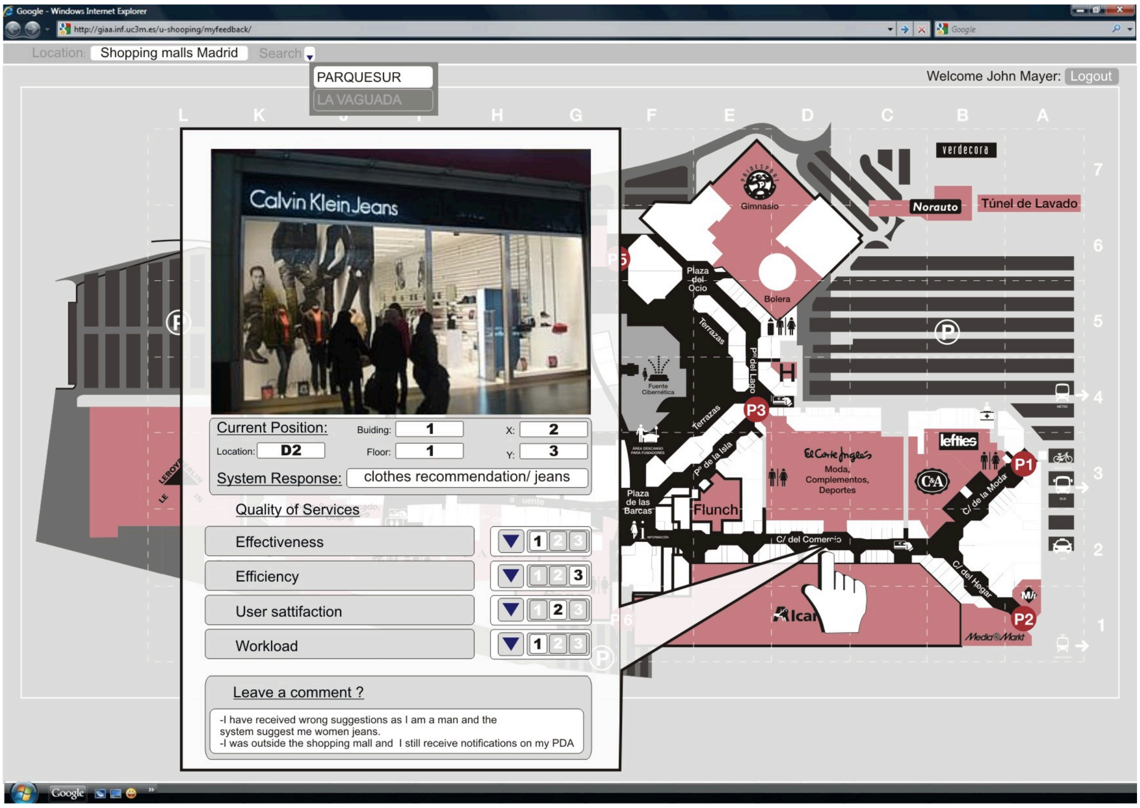


Figure 4.19: Online (left) and Offline (right) quality evaluation

coordinates. Finally, the system response is shown to the customer as a text message (i.e. "jeans recommendation"), so he can then proceed to fill in the quality parameters according to the system response (see Figure 4.19).

If $P_i1 =$ "*effectiveness*", $P_i2 =$ "*efficiency*", $P_i3 =$ "*usergeneralsatisfaction*", $P_i4 =$ "*workload*"; identified as quality parameters $Q_i = (P_i1, P_i2, P_i3, P_i4)$ and the vendor agent ranking for this consult is $R_i = (R_1, R_2, ..., R_9)$ given by the customer. Then, we can calculate the distance between the customer and the system The frequencies of the Manhattan distance, corresponding to the broker agent results for the satisfactory cases as well as the unsatisfactory cases, can be analysed to draw some conclusions and give this feedback to the system.

5 Conclusions

5.1 Final remarks

Ambient Intelligence and Smart Environments (SmE) emphasize on greater user friendliness, more efficient services support, user-empowerment, and support for human interactions. In this vision, people will be surrounded by intelligent and intuitive interfaces embedded in everyday objects around us and an environment recognising and responding to the presence of individuals in an invisible way. Although there are already solutions that have been successfully implemented to deliver the right service to the right user at the right time, there are several important issues not tackled in these kind of approaches: generic architecture,the self-adaptation and the user-centric evaluation paradigm. Due to highly dynamic properties of the above introduced environments, the software system running on them has to face problems such as: user mobility, service failure, resources or goal changes which may happen in any moment. To cope with these problems, such system must senses the environment, and acts on it, over time in pursuit of its own benefit.

Interest in research into context-aware systems has increased lately, since they are becoming one of the main new challenges of computer science, and particularly of artificial intelligence. So four objectives were put forward in order to design, develop and evaluate context-aware systems in AmI environments. According to this the first objective of this thesis that was fulfilled was to design and developed a centralized system for AmI environments. In order to do so, we use of a commercial platform (Appear) with a layered architecture allow us to manage the contextual information by means of a context manager included in the architecture. As applications are expected to provide a quality of service to users, they have to be able to adapt to context changes. The Appear Context Engine is a good context management solution as it provides for running a KBS on dynamically updated information about user location and statically defined information about the user profile. In our context aware system, the different functionalities are distributed into specific modules to guarantee the most efficient and adapted service to the user. Two subsystems have been defined that carry out sensing and rendering functions, context-aware web services management, context information is captured, updated, managed and stored by means of the interaction between the different modules in the three layers. The Appear IQ platform has been integrated for the web services management, dealing with the different processes that are required to implement this layer.

The formal representation of this contextual information was conceptualized by means of a knowledge based system (KBS) using Gaia methodology. One of the contributions is the definition of a KBS for a realistic scenario according to the CKADS methodology and conceptual modelling language (CML) in order to formally represent contextual information for the Appear platform. Linking a methodological approach (CKADS) to the development of the knowledge base while this knowledge base is integrated in a commercial, extensively used platform (Appear) is an innovative and relevant contribution to the state of the art.

Second objective was to develop a generic architecture using intelligent agents that can be applied for any scenario in AmI. For a large class of the envisaged AmI applications, the added value of new services is likely to be for people in ordinary social contexts. Such applications beg for technologies that are transparent, so that their functional behaviour can be easily understood by users. Intelligent agents seem to be the appropriate solution for AmI environments since they provide autonomy and proactivity. The design and implementation of the system based on agents is maximized using Gaia methodology (Franco Zambonelli & Wooldridge, 2003). An ontology based approach is used to represent the knowledge of the domain.

First, we defined the domain knowledge layer as a set of twenty two concepts with their respective attributes and relationships. Additionally, we modelled the set of production rules as an inference structure that deals dynamically with the system input/output behaviour. In this inference layer we described how the concepts and relationships are assigned to declarative specifications of dynamic roles. Finally, these inferences were fired in a sequential order defined by a control structure corresponding to one of the task templates suggested by CKADS. Particularly, we consider our problem to be a special case of an assessment task template, where an assessment problem consists of finding a decision category for a case based on domain-specific norms. Next we showed step-by-step how Appear was effectively configured to implement the KBS designed according to CKADS principles.

Third objective consists of including a multimodal user interface in order to ensure natural and intelligent interaction. In AmI it takes an important role because users can naturally interact with the system and thus perceive it as intelligent. To ensure such a natural and intelligent interaction, it is necessary to provide an effective, easy, save and transparent interaction between the user and the system. So, a multimodal dialogue system was introduced for the centralized system as well as for the distributed system. The scheme used for the development of these systems includes several generic modules that deal with multiple knowledge sources and that must cooperate to satisfy users requirements.

Fourth objective deals with the evaluation of AmI, not only from the software developer's point of view but also from the user's point of view. So here, we propose an evaluation methodology combining three steps (purpose, design and execution) relying in a taxonomy of pervasive computing systems studied in literature. We argue that due to highly dynamic properties of the AmI environments, it should exist a methodology for evaluating these systems taking into account the type of scenarios. We focus on the user-centred evaluation and use some metrics defined by user-centered parameters, as presented in Chapter 3. Finally a set of different scenarios of AmI were proposed. We provide the user with a toolkit to provide feedback so we can collect this information for being used later in the adaptation process. The system adapt not only to changes in the environment but also to the user requirements and needs. In our system, users take the relevant role providing an evaluation of the system behaviour while using it or once it has been used (online/ offline: OnE/ OffE).

5.2 Future work

Nowadays, there are open problems in the way Artificial Intelligence (AI) can contribute to Ambient Intelligence. Although some researches exist, there is still work to be done in Machine Learning field to collaborate in more adaptive systems. One aspect very important for AmI is the need to learn from user observation. This was our first motivation but we started instead designing, developing and proposing a method for evaluating context-aware systems in AmI, so adaptation process was uncovered in this thesis.

AI community also pay attention to Ontologies and Semantic Web. The early experience in Intelligent Systems development show us that intelligence is not possible without knowledge. However, little attention has been paid to Knowledge Representation in most of the Ambient Intelligence projects. Areas like Information Retrieval and Text Mining are also in the spotlight of researcher and although we have applied ontologies based model to our work it would be interesting to work on Semantic analysis.

Real-world problems are affected with incompleteness and uncertainty. Generally we deal with information, some part of this information is correct, some part may be in- correct, and some part is missing. The question is how to proceed with an elaborated reasoning process dealing with these information problems. Many techniques have been used but for us it is very interesting an approach to Bayesian Networks, Fuzzy Logic, Rough Sets to handle the problem. Since AmI environments are real, we are sure that Incompleteness and Uncertainty will be present there, and users expect support from these environments even if these problems exist.

Also, we think community needs an extra effort in developing more transparent techniques applied to the context-aware systems can interact with the user by means of multimodal user interfaces. We have develop a dialogue system using Speech Recognition and Natural Language but more complex interfaces can be developed focuses on challenges that designers face in creating interfaces for users of various virtual environments, applying for example Computer Vision techniques (Image Acquisition, Image Processing, Object Recognition (2D and 3D), Scene Analysis, and Image Flow Analysis.

Additionally since obtaining experimental results from a comparison of context-aware systems is not yet possible (current context-aware systems are too adhoc different to be implemented in a single testing domain), implementing a testbed for experimentation in context-aware systems is also a future work for us, where experimental results can be obtained from applying different metrics propose in this thesis for each type of scenario.

A

Published Results

This appendix lists the scientific works that have to do with this thesis organized by class of publication.

Articles

- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. "A Knowledge-Based System Approach for a Context-Aware System". Knowledge-based Systems Journal. 2011. JCR Impact factor: 1.525. *Accepted—in press.*
- Nayat Sánchez-Pi, Eleni Mangina, Javier Carbó and José M. Molina. "A Utility-Based Adaptation Approach for an AmI (Ambient Intelligence) Environment". International Journal of Artificial Intelligence. IJAI. Vol 6. S11. 2011. 100-111.
- Nayat Sánchez-Pi and José M. Molina. "A multi-agent approach for provisioning of e-services in u-commerce environments". Journal of Internet Research, Vol 3, Issue 2. 276-295. 2010. JCR Impact Factor: 1.150.
- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. "Analysis and Design of a Multi-Agent System Using Gaia Methodology in an Airport Case of Use". Revista Iberoamericana de Inteligencia Artificial. Vol 14, Nº45. 9-17. 2010.
- David Griol, Nayat Sánchez-Pi, Javier Carbó and José M. Molina. "A Context-Aware Architecture to Provide Spoken Access to Web Services". Knowledge–based Systems Journal. 2011. JCR Impact factor: 1.525. *Under review*
- David Griol, Nayat Sánchez-Pi, Javier Carbó and José M. Molina. "An Automatic Dialog Simulation Technique to Develop and Evaluate Interactive Conversational Agents". Mutiagent and Grid Systems Journal. 2011. *Under review*

Book Chapters

- Nayat Sánchez-Pi, David Griol, Javier Carbo, and Jose M. Molina. Evaluating Interaction of MAS Providing Context-Aware Services. Agents and Multi-Agents Systems:

Technologies and Applications. 5th KES International Conference, KES-AMSTA 2011. Lecture Notes in Artificial Intelligence. LNAI. 6682. 373-382. Springer Verlag.

- David Griol, Nayat Sánchez-Pi, Javier Carbo, and Jose M. Molina. An Agent-Based Dialog Simulation Technique to Develop and Evaluate Conversational Agents. Advances on Practical Applications of Agents and Multiagent Systems, 9th Int. Conference on Practical Applications of Agents and Multiagent Systems. 2011. Advances in Intelligent and Soft Computing, vol. 88, 255-264. 2011. (ISBN: 978-3-642-19874-8, ISSN: 1867-5662, Demazeau, Y.; Pechoucek, M.; Corchado, J.M.; Bajo Pérez, J. Eds.). cs-conference ranking: Position 52 among 701 considered (in the Artificial Intelligence field)

- Nayat Sánchez-Pi and Jose M. Molina. Adaptation of an Evaluation System for e-Health Environments. Lecture Notes in Artificial Intelligence. Computer Science, Springer, LNAI 6279. Part IV. 357-364. 2010.

- David Griol, Nayat Sánchez-Pi, Javier Carbo, and Jose M. Molina. An Architecture for the Design of Context-Aware Conversational Agents. Advances in Practical Applications of Agents and Multiagent Systems. 41-46. 2010.

- Nayat Sánchez-Pi, Eleni Mangina, Javier Carbó and José M. Molina. Multi-Agent System (MAS) Applications in Ambient Intelligence (AmI) Environments. Springer-Verlag. Trends in Practical Applications of Agents and Multiagent Systems. 493-500. 2010.

- Nayat Sánchez-Pi and José M. Molina. A Centralized Approach to an Ambient Assisted Living Application: An Intelligent Home. Springer Lecture Notes in Computer Science. LNCS 5518. Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living. 706-709. 2009.

- Nayat Sánchez-Pi and José M. Molina. A Smart Solution for Elders in Ambient Assisted Living. Springer Lecture Notes in Computer Science. LNCS 5601 Methods and Models in Artificial and Natural Computation. 415-425. 2009.

- Virginia Fuentes, Nayat Sánchez-Pi, Javier Carbo, and Jose M. Molina. Designing a Distributed Context–Aware Multi–Agent System" from Agent-Based Ubiquitous Computing. Agent-based Ubiquitous Computing, Series: Atlantis Ambient and Pervasive Intelligence - Vol. 1. E. Mangina, J. Carbó, J.M.Molina (Eds.). Ed. World Scientific. ISBN 978-90-78677-10-9. 117-130. 2009.

- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. Building a Knowledge Based System for an Airport Case of Use. Advances in Soft Computing. DCAI 2008, Springer-Verlag Berlin Heidenlberg. ASC 50. ISBN: 978-3-540-85862-1. 739-747. 2008.

- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. JADE/LEAP Agents in an Ambient Intelligence Domain. Lecture Notes Artificial Intelligence. LNAI 5271, Springer-Verlag Berlin Heidelberg. 62-69. 2008.

- Nayat Sánchez-Pi, Co-author of: José M. Molina, Juan M. Corchado and Javier Bajo. Ubiquitous Computing for Mobile Environments. Issues in Multi-Agent Systems. The

AgentCities.ES Experience. Whitestein Series in Software Agent Technologie. Birkäuser Verlag Basel/ Switzerland. 2007. ISBN: 978-3-7643-8542-2. 33-57. 2007.

Conferences Publications

- David Griol, Nayat Sánchez-Pi, Javier Carbo, and Jose M. Molina. Context-Aware Approach for Orally Accessible Web Services. Advances on Practical Applications of Agents and Multiagent Systems. IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology. WI-IAT, vol. 3, 171-174. Milán. Italy.
- David Griol, Nayat Sánchez-Pi, Javier Carbo, and Jose M. Molina. A Context-Aware Architecture to Provide Adaptive Services by means of Spoken Dialogue Interaction. International Conference on Artificial Intelligence. ICAI vol. 2. 912-918. ICAI 2009, Las Vegas Nevada, USA.
- Nayat Sánchez-Pi and Jose M. Molina. A Multi-Agent Platform for the Provisioning of U-commerce Services. 28th North American Fuzzy Information Processing Society Annual Conference (NAFIPS 2009). 1-6. Cincinnati, USA. June 2009.
- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. Analysis and Design of a Multi-Agent System Using Gaia Methodology in an Airport Case of Use. IBERAGENTS 2008. 7th Ibero-American Workshop in Multi-Agent Systems. In Conjunction with IBERAMIA'08. Lisbon. Portugal. 81-92. 2008.
- Nayat Sánchez-Pi, Javier Carbó and José M. Molina. Building a Knowledge Based System for an Airport Case of Use. Second International Workshop on User Centric Technologies and Applications (MADRINET). In Conjunction with CAEPIA'08. Salamanca. 98-106. 2008.
- Nayat Sánchez-Pi and Jose M. Molina. A Case Study of Context-Aware Services Domain Using a Commercial Wireless Location Sensing Technology and a Commercial Platform. 1st International Workshop on User Centric Technologies and Applications (MADRINET). In Conjunction with CAEPIA'07. 12-21. Salamanca.2007.
- Nayat Sánchez-Pi, Virginia Fuentes, Javier Carbó and José M. Molina. Knowledge-based system to define context in commercial applications. IEEE. 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD2007) IEEE. Qingdao. China. 694-699. 2007.
- Virginia Fuentes, Nayat Sánchez-Pi, Javier Carbó and José M. Molina. Generic Context-Aware BDI Multi-Agent Framework with GAIA methodology. Agent-Based Ubiquitous Computing (ABUC) Workshop in 2007 International Conference on Autonomous Agents and Multi-Agent Systems AAMAS '07. Hawaii. USA. 694-699. 2007.
- Virginia Fuentes,Nayat Sánchez-Pi, Javier Carbó and José M. Molina. Reputation in User Profiling in a Context-aware Multi-Agent System. Fourth European Workshop on Multi-Agent Systems EUMAS'06. Lisbon. Portugal. 2006.

References

Aarts, E., Harwig, E., & Schuurmans, M. (2001). *Ambient Intelligence in Denning, J.(edt.): The Invisible Future*. McGraw-Hill, New York.

Abdualrazak, B., Malik, Y., & Yang, H.-I. (2010). A taxonomy driven approach towards evaluating pervasive computing system. In Y. Lee, Z. Bien, M. Mokhtari, J. Kim, M. Park, J. Kim, H. Lee, & I. Khalil (Eds.) *Aging Friendly Technology for Health and Independence*, vol. 6159 of *Lecture Notes in Computer Science*, (pp. 32–42). Springer.

Abowd, G., Atkeson, C., Bobick, A., Essa, I., MacIntyre, B., Mynatt, E., & Starner, T. (2000). Living laboratories: the future computing environments group at the Georgia Institute of Technology. In *CHI'00 extended abstracts on Human factors in computing systems*, (pp. 215–216).

Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R., & Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, *3*(5), 421–433.

Afsarmanesh, H., Guevara, M., & Hertzberger, L. (2004). Virtual community support in telecare. *L. Camarinha-Matos & H. Afsarmanesh (Eds.) Processes and Foundations for Virtual Organizations*, (pp. 211–220).

Agarawala, A., Greenberg, S., & Ho, G. (2004). The context-aware pill bottle and medication monitor. In *Video Proceedings and Proceedings Supplement of the UBICOMP 2004*.

Akman, V., & Surav, M. (1997). The use of situation theory in context modeling. *Computational Intelligence*, *13*(3), 427–438.

Andreani, G., Di Fabbrizio, G., Gilbert, M., Gillick, D., Hakkani-Tur, D., & Lemon, O. (2006). Let's discoh: collecting an annotated open corpus with dialogue acts and reward signals for natural language helpdesks. *IEEE/ACL Spoken Language Technology*.

Angele, J., Fensel, D., & Studer, R. (1996). *Domain and task modeling in MIKE*. Univ.

Appear Networks (2008). Appear.
URL `http://www.appearnetworks.com`

Ardissono, L., Furnari, R., Goy, A., Petrone, G., & Segnan, M. (2007). Context-aware workflow management. *Web Engineering*, (pp. 47–52).

Aruba Networks (2007). Aruba.
URL `http://www.arubanetworks.com`

Asif, Z., & Mandviwalla, M. (2005). Integrating the supply chain with rfid: A technical and business analysis. *Communications of the Association for Information Systems*, *5*(1), 24.

Ayed, D., Delanote, D., & Berbers, Y. (2007). Mdd approach for the development of context-aware applications. *Modeling and Using Context*, (pp. 15–28).

Bacon, J., Bates, J., & Halls, D. (1997). Location-oriented multimedia. *Personal Communications, IEEE*, *4*(5), 48–57.

Bahl, P., & Padmanabhan, V. (2000). Radar: An in-building rf-based user location and tracking system. In *NFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, (pp. 775–784). IEEE.

Bajo, J., Julian, V., Corchado, J., Carrascosa, C., Paz, Y. D., Botti, V., & Paz, J. D. (2008). An execution time planner for the artis agent architecture. *Journal of Engineering Applications of Artificial Intelligence*, *21*(8).

Baldauf, M., Dustdar, S., & Rosenberg, F. (2007). A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, *2*(4), 263–277.

Barton, J., & Vijayaraghavan, V. (2002). Ubiwise, a ubiquitous wireless infrastructure simulation environment. *HP Labs*.

Barwise, J., & Perry, J. (1981). Situations and attitudes. *The Journal of Philosophy*, *78*(11), 668–691.

Bauer, J., Kutsche, R., & Ehrmanntraut, R. (2003). Identification and modeling of contexts for different information scenarios in air traffic. *Technische Universitat Berlin, Diplomarbeit*.

Bayeh, R., Lin, S., Chollet, G., & Mokbel, C. (2004). Towards multilingual speech recognition using data driven source/target acoustical units association. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 1, (pp. i–521).

Becker, M., Werkman, E., Anastasopoulos, M., & Kleinberger, T. (2006). Approaching ambient intelligent home care system. In *Pervasive Health Conference and Workshops 2006*, (pp. 1–10).

Bellotti, V., Back, M., Edwards, W. K., Grinter, R. E., Henderson, A., & Lopes, C. (2002). Making sense of sensing systems: five questions for designers and researchers. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, CHI '02, (pp. 415–422). New York, NY, USA: ACM.
URL `http://doi.acm.org/10.1145/503376.503450`

Berners-Lee, T., & Miller, E. (2002). The semantic web lifts off. *ERCIM News*, *51*, 9–11.

Bernsen, N., Dybkjaer, L., & Dybkjaer, H. (1994). A dedicated task-oriented dialogue theory in support of spoken language dialogue system design. In *Proc. of the 3rd International Conference on Spoken Language Processing (ICSLP'94)*, (pp. 875–878). Yokohama (Japan).

Biegel, G., & Cahill, V. (2004). A framework for developing mobile, context-aware applications. In *Second IEEE Annual Conference on Pervasive Computing and Communications*.

Bohus, D., Grau, S., Huggins-Daines, D., Keri, V., Krishna, G., Kumar, R., Raux, A., & Tomko, S. (2007a). Conquest - an Open-Source Dialog System for Conferences. In *Proc. of Human Language Technologies (HLT/NAACL'07)*, (pp. 9–12). Rochester, NY, USA.

Bohus, D., Puerto, S. G., Huggins-Daines, D., Keri, V., Krishna, G., Kumar, R., Raux, A., & Tomko, S. (2007b). Conquest: An open-source dialog system for conferences. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers on XX*, (pp. 9–12).

Bohus, D., & Rudnicky, A. (2005). LARRI: A language-based maintenance and repair assistant. *Spoken multimodal human-computer dialogue in mobile environments*, (pp. 203–218).

Bonneau-Maynard, H., Devillers, L., & Rosset, S. (2000). Predictive performance of dialog systems. In *Language Resources and Evaluation Conference*.

Bopp, K. (1990). How patients evaluate the quality of ambulatory medical encounters: a marketing perspective. *Journal of Health Care Marketing*, *10*(1), 6–15.

Bouwman, G., Sturm, J., & Boves, L. (1999). Incorporating confidence measures in the dutch train timetable information system developed in the arise project. In *icassp*, (pp. 493–496).

Bouzy, B., & Cazenave, T. (1997). Using the object oriented paradigm to model context in computer go. In *Proceedings of the First International and Interdisciplinary Conference on Modeling and Using Context*.

Bratman, M. (1987). *Intentions, Plans and Practical Reasoning*. Cambridge, Massachusetts: Harvard University Press.

Bratman, M. (1999). Intention, plans, and practical reason. Cambridge University Press.

Bratman, M., Israel, D., & Pollack, M. (1998). Plans and resource-bounded practical reasoning. *Computational Intelligence*, *4*(4), 349–355.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., & Mylopoulos, J. (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, *8*(3), 203–236.

Brown, P. (1996). The stick-e document: A framework for creating context-aware applications. In *Proceedings published in Origination, Dissemination, and Design (EP-ODD), Palo Alto, CA,*, vol. 8, (pp. 259–272). Chichester: John Wiley and Sons.

Brown, P., Bovey, J., & Chen, X. (1997). Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, *4*(5), 58–64.

Buchholz, T., Küpper, A., & Schiffers, M. (2003). Quality of context: What it is and why we need it. In *Proceedings of the Workshop of the HP OpenView University Association*, (pp. 1–13).

Burke, R., Hammond, K., & Young, B. (1996). Knowledge-based navigation of complex information spaces. In *Proceedings of The National Conference On Artificial Intelligence*, vol. 462, (p. 468).

Burrell, J., & Gay, G. (2002). E-graffiti: evaluating real-world use of a context-aware system. *Interacting with Computers – Special Issue on Universal Usability*, *14*(4), 301–312.

Busetta, P., Ronnquist, R., Hodgson, A., & Lucas, A. (1999). Jack intelligent agents-components for intelligent agents in java. *AgentLink News Letter*, *2*, 2–5.

Bylund, M., & Espinoza, F. (2002). Testing and demonstrating context-aware services with quake iii arena. *Communications of the ACM*, *45*(1), 46–48.

Capra, L., Emmerich, W., & Mascolo, C. (2001). Reflective middleware solutions for context-aware applications. *Metalevel Architectures and Separation of Crosscutting Concerns*, (pp. 126–133).

Carroll, J., Olson, J., & Anderson, N. (1987). *Mental models in human-computer interaction: Research issues about what the user of software knows*. National Academies.

Chen, H. (2004). *An Intelligent Broker Architecture for Pervasive Context-Aware Systems.*. Ph.D. thesis, University of Maryland, Baltimore County.

Chen, H., Finin, T., & Joshi, A. (2003). Using owl in a pervasive computing broker. In *Workshop on Ontologies in Agent Systems*.

Chen, P. (1976). The entity-relationship model—toward a unified view of data. *ACM Transactions on database systems*, *1*(1), 9–36.

Cheverst, K., Davies, N., Mitchell, K., Friday, A., & Efstratiou, C. (2000). Developing a context-aware electronic tourist guide: some issues and experiences. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems, ACM Press, New York, USA*, (pp. 17–24). New York, USA: ACM Press.

Cheverst, K., Mitchell, K., & Davies, N. (1999). Design of an object model for a context sensitive tourist guide. *Computers and Graphics*, *23*(6), 883–891.

Chiang, F., Braun, R. M., Magrath, S., Markovits, S., & Huang, S. (2005). Autonomic service configuration in telecommunication mass with extended role-based gaia and jadex. In *IEEE International Conference on Services Systems and Services Management*, (pp. 1319–1324).

Chtcherbina, E., & Franz, M. (2003). Peer-to-peer coordination framework (p2pc): Enabler of mobile ad-hoc networking for medicine, business, and entertainment. In *Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet*.

Cole, R., Mariani, J., Uszkoreit, H., Varile, G. B., Zaenen, A., Zampolli, A., & Zue, V. (1997). *Survey of the State of the Art in Human Language Technology*. Cambridge University Press.

Corchado, J., & Laza, R. (2003). Constructing deliberative agents with case-based reasoning technology. *International Journal of Intelligent Systems*, *18*(12), 1227–1241.

Córdoba, R., San-Segundo, R., Montero, J., Colás, J., Ferreiros, J., Macías-Guarasa, J., & Pardo, J. (2001). An interactive directory assistance service for spanish with large-vocabulary recognition. In *Proceedings of Eurospeech*, (pp. 1279–1282).

Coutaz, J., Crowley, J. L., Dobson, S., & Garlan, D. (2005). Context is key. *Communications of the ACM*, *48*(3), 49–53.

Cronin Jr, J., & Taylor, S. (1992). Measuring service quality: a reexamination and extension. *The Journal of Marketing*, *56*(3), 55–68.

Cuayahuitl, H., Renals, S., Lemon, O., & Shimodaira, H. (2005). Human-computer dialogue simulation using hidden Markov models. In *2005 IEEE Workshop on Automatic Speech Recognition and Understanding*, (pp. 290–295).

Dahlback, N., Jonsson, A., & Ahrenberg, L. (1993). Wizard of oz studies-why and how. *Knowledge-Based Systems*, *6*(4), 258–266.

Davidsson, P., Johansson, S., & Svahnberg, M. (2006). Characterization and evaluation of multi-agent system architectural styles. *Software Engineering for Multi-Agent Systems IV*, (pp. 179–188).

De Bruijin, J. (2003). Using ontologies — Enabling knowledge sharing and reuse on the semantic web. Tech. rep., Technical Report DERI-2003-10-29, Digital Enterprise Research Institute (DERI), Austria.

DeLoach, S. (1999). Multiagent systems engineering: A methodology and language for designing agent systems. In *Proceedings of Agent-Oriented Information Systems (AOIS) '99*.

DeLoach, S. (2004). The MASE methodology. *Methodologies and software engineering for agent systems*, (pp. 107–125).

Dennett, D. (1989). *The intentional stance*. The MIT Press.

Dey, A. (2001). Understanding and using context. personal ubiquitous computing. *5*(1), 4–7.

Dey, A., & Abowd, G. (1999). The context toolkit: Aiding the development of context-enabled applications. In *Proc. of the SIGCHI conference on Human factors in computing systems (CHI'99)*, (pp. 434–441). Pittsburgh, USA.

Dey, A., & Abowd, G. (2000). Towards a better understanding of context and context-awareness. In *Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness*. New York.: ACM Press.

Dey, A., Mankoff, J., Abowd, G., & Carter, S. (2002). Distributed mediation of ambiguous context in aware environments. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, (pp. 121–130).

D'Inverno, M., Luck, M., Georgeff, M., Kinny, D., & Wooldridge, M. (2004). The DMARS architecture: A specification of the distributed multi-agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, *9*(1), 5–53.

Domingue, J., Motta, E., & Watt, S. (1993). *The emerging VITAL workbench*, (pp. 320–320). Springer.

Durfee, E., & Rosenschein, J. (1994). Distributed problem solving and multi-agent systems: Comparisons and examples. In *13th International Workshop on Distributed Artificial Intelligence*, (pp. 94–104).

Ekahau (2011). Ekahau web page.
URL http://www.ekahau.com/

Emiliani, P., & Stephanidis, C. (2005). Universal access to ambient intelligence environments: Opportunities and challenges for people with disabilities. *IBM Systems Journal*, *44*(3), 605–619.

Eriksson, H., Shahar, Y., Tu, S., Puerta, A., & Musen, M. (1995). Task modeling with reusable problem-solving methods. *Artificial Intelligence*, *79*(2), 293–326.

Espinoza, F., Persson, P., Sandin, A., Nystrom, H., Cacciatore, E., & Bylund, M. (2001). Geonotes: social and navigational aspects of location-based information systems. In *Proceedings of the 3rd International Conference on Ubiquitous Computing*, (pp. 2–17). Atlanta, Georgia, USA.

Esteve, Y., Raymond, C., Bechet, F., & Mori, R. (2003). Conceptual decoding for spoken dialog systems. In *Eighth European Conference on Speech Communication and Technology*.

Fahy, P., & Clarke, S. (2004). Cass – a middleware for mobile context-aware applications. In *Workshop on Context-Awareness, MobiSys 2004*.

Federal Communications Commission (2001). FCC wireless 911 requirements fact sheet.
URL http://www.fcc.gov/e911/

Floeck, M., & Litz, L. (2007). Integration of home automation technology into an assisted living concept. *Assisted Living Systems-Models, Architectures and Engineering Approaches*.

Forbes-Riley, K., & Litman, D. (2006). Modelling user satisfaction and student learning in a spoken dialogue tutoring system with generic, tutoring, and user affect parameters. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, (pp. 264–271).

Fraile, J., Bajo, J., & Corchado, J. (2008). Amade: Developing a multi-agent architecture for home care environments. In *7th Ibero-American Workshop in Multi-Agent Systems*.

Franco Zambonelli, N. R. J., Reggio Emilia, & Wooldridge, M. (2003). Developing multiagent systems: The gaia methodology. *ACM Transactions on Software Engineering and Methodology*, *12*, 317–370.

Friedman, B., & Kahn Jr, P. (2002). Human values, ethics, and design. In *The human-computer interaction handbook*, (pp. 1177–1201). L. Erlbaum Associates Inc.

Fuentes, V., Carbó, J., & Molina, J. M. (2006a). Heterogeneous domain ontology for location based information system in a multi-agent framework. In *IDEAL*, (pp. 1199–1206).

Fuentes, V., Sanchez-Pi, N., Carbo, J., & Molina, J. M. (2006b). Reputation in user profiling for a context-aware multiagent system. In *European Conference on Multi-agent Systems (EUMAS)*.

García, F., Hurtado, L., Sanchis, E., & Segarra, E. (2003). The incorporation of confidence measures to language understanding. In *Text, Speech and Dialogue*, (pp. 165–172).

Gaspar, G. (1991). Communication and belief changes in a society of agents: Towards a formal model of autonomous agent, d.a.i. 2, 1991. In Y. Demazeau, & J. P. Muller (Eds.) *Descentralized A. I. 2*, (pp. 245–255). Amsterdam: Elsevier Science.

Genesereth, M. R., & Ketchpel, S. P. (1994). Software agents. *Communications of ACM*, *37*(7), 48–53.

Georgeff, M., & Lansky, A. (1987). Reactive reasoning and planning. In *Proceedings of the sixth national conference on artificial intelligence (AAAI-87)*, (pp. 677–682).

Georgeff, M., Pell, B., Pollack, M., Tambe, M., & Wooldridge, M. (1999). The belief-desire-intention model of agency. *Intelligent Agents V: Agents Theories, Architectures, and Languages*, (pp. 1–10).

Ghidni, C., & Giunchiglia, F. (2001). Local models semantics, or contextual reasoning locality compatibility. *Artificial Intelligence*, *127*(2), 221–259.

Giunchiglia, F. (1993). Contextual reasoning. epistemologica. Tech. rep., Also IRST-Technical Report 9211-20, IRST, Trento, Italy.

Giunchiglia, F., Mylopoulos, J., & Perini, A. (2002). The tropos software development methodology: Processes, models and diagrams. In *Proceedings of the first international joint conference on autonomous agents and multiagent systems*, (pp. 63–74). ACM press.

Glass, J., Flammia, G., Goodine, D., Phillips, M., Polifroni, J., Sakai, S., Seneff, S., & Zue, V. (1995a). Multilingual spoken-language understanding in the MIT Voyager system. *Speech Communication*, *17*, 1–18.

Glass, J., Flammia, G., Goodine, D., Phillips, M., Polifroni, J., Sakai, S., Seneff, S., & Zue, V. (1995b). Multilingual spoken-language understanding in the mit voyager system. *Speech Communication*, *17*(1-2), 1–18.

Goker, A., & Myrhaug, H. (2002). User context and personalisation. In *Proceedings of the European Conference on Case Based Reasoning (ECCBR 2002)-Workshop on Personalized Case-Based Reasoning, Aberdeen, Scotland*, (pp. 4–7).

Gomez-Perez, A., & Benjamins, V. (1999). Overview of knowledge sharing and reuse components: ontologies and problem-solving methods. In *IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5)*. Stockholm.

Gray, P., & Salber, D. (2001). Modelling and using sensed context information in the design of interactive applications. *Engineering for Human-Computer Interaction*, (pp. 317–335).

Griol, D., Hurtado, L., Sanchis, E., & E., S. (2009a). Acquiring and evaluating a dialog corpus through a dialog simulation technique. In *Proc. of the 9th SIGdial Workshop on Discourse and Dialogue*, (pp. 326–332). London, UK.

Griol, D., Hurtado, L. F., Segarra, E., & Sanchis, E. (2008). A statistical approach to spoken dialog systems design and evaluation. *Speech Communication*, *50*, 666–682.

Griol, D., Sánchez-Pi, N., Carbó, J., & Molina, J. M. (2009b). Context-aware approach for orally accessible web services. In *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology. WI-IAT*. Milan. Italy.

Griol, D., Sánchez-Pi, N., Carbó, J., & Molina, J. M. (2010). An architecture for the design of context-aware conversational agents. In Y. Demazeau, F. Dignum, J. M. Corchado, & J. Bajo Pérez (Eds.) *Advances in Practical Applications of Agents and Multiagent Systems*, Advances in Intelligent and Soft Computing, (pp. 41–46). Salamanca (Spain): Springer-Verlag.

Gronroos, C. (1990). *Service management and marketing: managing the moments of truth in service competition*. Jossey-Bass.

Gross, T., & Specht, M. (2001). Awareness in context-aware information systems. In K. e. Oberquelle, Oppermann (Ed.) *Proceedings of the Mensch und Computer - 1*, (pp. 173–182). Bad Honnef, Germany: Fachübergreifende Konferenz.

Gruber, T. (1991). The role of common ontology in achieving sharable, reusable knowledge bases. In *2nd International Conference on Principles of Knowledge Representation and Reasoning*, (pp. 601–602). Cambridge, USA.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, *5*(2), 199–220.

Gruber, T. (1995). Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human Computer Studies*, *43*(5), 907–928.

Gruninger, M., & Fox, M. (1995). Methodology for the design and evaluation of ontologies. In *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI)*, vol. 95.

Gu, T., & Pung, H. (2004). A middleware for building context-aware mobile services. In *Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th*, vol. 5, (pp. 2656–2660).

Guanling Chen, M. L., & Kotz, D. (2004). Design and implementation of a large-scale context fusion network. In *Proc. of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, (pp. 246–255). Boston, USA.

H. Lee, J. H., J. Kim (2006). Context-aware based mobile service for ubiquitous home. In *8th Int. Conf. Advanced Communication Technology*, vol. 3.

Halpin, T., Morgan, A., & Morgan, T. (2008). *Information modeling and relational databases*. Morgan Kaufmann.

Hardy, H., Biermann, A., Inouye, R., McKenzie, A., Strzalkowski, T., Ursu, C., Webb, N., & Wu, M. (2006). The amitiè system: Data-driven techniques for automated dialogue. *Speech Communication*, *48*(3–4), 354–373.

Harter, A., Hopper, A., Steggles, P., Ward, A., & Webster, P. (2002). The anatomy of a context-aware application. *Wireless Networks*, *8*(2), 187–197.

Hassel, L., & Hagen, E. (2005). Adaptation of an automotive dialogue system to users' expertise. In *6th SIGdial Workshop on Discourse and Dialogue*.

Haux, R. (2006). Health information systems-past, present, future. *International Journal of Medical Informatics*, *75*(3-4), 268–281.

He, Y., & Young, S. (2003). A data-driven spoken language understanding system. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, (pp. 583–588).

Heim, J., Nilsson, E., & Skjetne, J. (2006). User profiles for adapting speech support in the opera web browser to disabled users. In *Proceedings of the 9th conference on User interfaces for all*, (pp. 154–172).

Helal, S., Giraldo, C., Kaddoura, Y., Lee, C., Zabadani, H. E., & Mann, W. (2003). Smart phone based cognitive assistant. In *Proc. UbiHealth 2003, Seattle, Washington*.

Held, A., Buchholz, S., & Schill, A. (2002). Modeling of context information for pervasive computing application. In *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI)*, vol. 22.

Henricksen, K., Indulska, J., & Rakotonirainy, A. (2002). Modeling context information in pervasive computing systems. *Pervasive Computing*, (pp. 79–117).

Henricksen, K., Indulska, J., & Rakotonirainy, A. (2003). Generating context management infrastructure from high-level context models. In *Proceedings of the 4th International Conference on Mobile Data Management (MDM'03)*, (pp. 1–6).

Hightower, J., & Borriello, G. (2001). A survey and taxonomy of location systems for ubiquitous computing. *IEEE Computer*, *34*(8), 57–66.

Hirschman, L. (2000). Evaluating spoken language interaction: Experiences from the darpa spoken language program 1990-1995. *Spoken Language Discourse. MIT Press, Cambridge, Mass*.

Hirschman, L., Bates, M., Dahl, D., Fisher, W., Garofolo, J., Pallett, D., Hunicke-Smith, K., Price, P., Rudnicky, A., & Tzoukermann, E. (1993). Multi-site data collection and evaluation in spoken language understanding. In *Proceedings of the workshop on Human Language Technology*, (pp. 19–24).

Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G., Altmann, J., & Retschitzegger, W. (2003). Context-awareness on mobile devices-the hydrogen approach. IEEE Computer Society.

Huber, M. J. (1999). JAM: A BDI-theoretic mobile agent architecture. In *Proceedings of the third annual conference on Autonomous Agents*, (pp. 236–243).

Huebscher, M., & McCann, J. (2004). Simulation model for self-adaptive applications in pervasive computing.

Hull, R., Neaves, P., & Bedford-Roberts, J. (1997). Towards situated computing. In B. e. Krulwich (Ed.) *The First International Symposium on Wearable Computers (ISWC '97), Cambridge, MA*.

Hurtado, L. F., Griol, D., Segarra, E., & Sanchis, E. (2006). A stochastic approach for dialog management based on neural networks. In *Proc. of Interspeech*, (pp. 49–52).

Hübner, J., & Bordini, R. (2010). Jason.
URL http://jason.sourceforge.net/

IEEE FIPA Standards Committee (1997). Foundations for intelligent phisical agents specification (FIPA).
URL http://www.fipa.org/

Indulska, J., Robinson, R., Rakotonirainy, A., & Henricksen, K. (2003). Experiences in using cc/pp in context-aware systems. In *Mobile Data Management*, (pp. 247–261).

Ingrand, F. F., Georgeff, M., & Rao, A. (1992). An architecture for real-time reasoning and system control. *IEEE EXPERT INTELLIGENT SYSTEMS and THEIR APPLICATIONS*, (pp. 34–44).

Jameson, A., Baldes, S., & Kleinbauer, T. (2003). Enhancing mutual awareness in group recommender systems. In B. Mobasher, & S. S. Anand (Eds.) *Proceedings of the IJCAI 2003 Workshop on Intelligent Techniques for Web Personalization*. Menlo Park, CA: AAAI. Available from http://dfki.de/~jameson/abs/JamesonBK03ITWP.html.

Jang, S., Ko, E., & Woo, W. (2005). Unified user-centric context: Who, where, when, what, how and why. *Personalized Context Modeling and Management for UbiComp Applications*, *149*(26-34).

Jo, C., Chen, G., & Choi, J. (2004). A new approach to the bdi agent-based modeling. In *Proceedings of the 2004 ACM symposium on Applied computing*, (pp. 1541–1545).

Johnson, C., & Turley, J. (2006). The significance of cognitive modeling in building healthcare interfaces. *International Journal of Medical Informatics*, *75*(2), 163–172.

Jonker, C., Klusch, M., & Treur, J. (2004). Design of collaborative information agents. *Cooperative Information Agents IV — The Future of Information Agents in Cyberspace*, (pp. 417–438).

Joumaa, H., Demazeau, Y., & Vincent, J. (2008). Evaluation of multi-agent systems: The case of interaction. In *3rd International Conference on Information and Communication Technologies: From Theory to Applications*, (pp. 1–6). IEEE.

Kagal, L., Korolev, V., Chen, H., Joshi, A., & Finin, T. (2001). Project centaurus: A framework for indoor services mobile services. In *Proceedings of the 21st International Conference on Distributed Computing Systems*.

Kang, D., Lee, H., Ko, E., Kang, K., & Lee, J. (2006). A wearable context aware system for ubiquitous healthcare. In *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, (pp. 5192–5195).

Kerer, C., Dustdar, S., Jazayeri, M., Gomes, D., Szego, A., & Caja, J. (2004). Presence-aware infrastructure using web services and rfid technologies. In *Proceedings of the 2nd European Workshop on Object Orientation and Web Services*. Oslo, Norway.

Kerttula, M., & Tokkonen, T. (2001). The total user experience-how to make it positive in future wireless systems and services. In *WWRF Annual Workshop in Paris*.

Kinny, D., Georgeff, M., & Rao, A. (1996). A methodology and modelling technique for systems of BDI agents. *Agents breaking away*, (pp. 56–71).

Kinny, D., & Phillip, R. (2004). Building composite applications with goal-directed agent technology. *AgentLink News*, *16*, 6–8.

Korel, B. T., & Kao, S. G. (2007). Addressing context awareness techniques in body sensor networks. In *21st International Conference on Advanced Information Networking and Applications Workshops 2007*, (pp. 798–803).

Kovács, G., & Kopácsi, S. (2006). Some aspects of ambient intelligence. *Acta Polytechnica Hungarica*, *3*(1), 35–60.

Krause, M., & Hochstatter, I. (2005). Challenges in modelling and using quality of context (qoc). *Mobility Aware Technologies and Applications*, (pp. 324–333).

Kurumatani, K., Sashima, A., & Izumi, N. (2004). Location-mediated web services coordination in ubiquitous computing. In *IEEE I nt. Conf. on Web Services (ICWS2004)*, (pp. 822–823).

Langner, B., & Black, A. (2005). Using speech in noise to improve understandability for elderly listeners. In *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*, (pp. 392–396).

Lech, T. C., & Wienhofen, L. W. M. (2005). Ambieagents: a scalable infrastructure for mobile and context-aware information services. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, (pp. 625–631). New York, NY, USA: ACM.

Lemon, O., Georgila, K., & Henderson, J. (2006). Evaluating effectiveness and portability of reinforcement learned dialogue strategies with real users: the talk towninfo evaluation. In *2006 IEEE Spoken Language Technology Workshop*, (pp. 178–181).

Lesser, V. (1999). Cooperative multiagent systems: A personal view of the state of the art. *IEEE Transactions on Knowledge and Data Engineering*, *11*(1), 133–142.

Levin, E., & Pieraccini, R. (1995). Concept-based spontaneous speech understanding system. In *Fourth European Conference on Speech Communication and Technology*.

Levin, E., Pieraccini, R., & Eckert, W. (2000). A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions onSpeech and Audio Processing*, *8*(1), 11–23.

Litman, D. J., & Silliman, S. (2004). ITSPOKE: An Intelligent Tutoring Spoken Dialogue System. In *Proc. of the Human Language Technology Conference (HLT/NAACL'04)*, (pp. 233–236). Boston, USA.

Little, M., Newcomer, E., & Pavlik, G. (2007). Web services context specification (WS-Context). Tech. Rep. Version 1.0, OASIS Committee.
URL http://docs.oasis-open.org/ws-caf/ws-context/v1.0/wsctx.pdf

López, D., & Mateo, C. (2005). Application of confidence measures for dialogue systems through the use of parallel speech recognizers. In *Ninth European Conference on Speech Communication and Technology*.

López-Cózar, R., & Araki, M. (2005). Spoken, Multilingual and Multimodal Dialogue Systems. In *John Wiley & Sons Publishers*.

López-Cózar, R., & Araki, M. (2005). *Spoken, Multilingual and Multimodal Dialogue Systems: Development and Assessment*. John Wiley Sons.

Lyytinen, K., & Yoo, Y. (2002). Research commentary: the next wave of nomadic computing. *Information Systems Research*, *13*(4), 377–388.

Lyytinen, K., Yoo, Y., Varshney, U., Ackerman, M., Davis, G., Avital, M., Robey, D., Sawyer, S., & Sorensen, C. (2004). Surfing the next wave: design and implementation challenges of ubiquitous computing environments. *Communications of the Association for Information Systems*, (pp. 697–716).

Maes, P. (1990a). Designing autonomous agents: theory and practice from biology to engineering and back. *MIT Press*.

Maes, P. (1990b). Situated agents can have goals. *Robotics and autonomous systems*, *6*(1-2), 49–70.

Mas, A. (2005). *Agentes software y sistemas multiagente: conceptos, arquitecturas y aplicaciones.*. Prentice Hall.

Mccarthy, J. (1993). Notes on formalizing contexts. In M. Kaufmann (Ed.) *In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, (pp. 555–560). San Mateo, California.

Mccarthy, J., & Buva, C. (1997). Formalizing context (expanded notes). In *Working Papers of the AAAI Fall Symposium on Context in Knowledge Representation and Natural Language*, (pp. 99–135). Menlo Park, California: American Association for Artificial Intelligence.

McGovern, E., Roche, B., Collier, R., & Mangina, E. (2007). Iumela: A lightweight multi-agent systems based mobile learning assistant using the abits messaging service. In *LNCS Ubiquitous Intelligence and Computing*, vol. 4611. Springer Berlin/Heidelberg: Springer Berlin/Heidelberg.

Melin, H., Sandell, A., & Ihse, M. (2001a). Ctt-bank: A speech controlled telephone banking system - an initial evaluation. In *TMH-QPSR*, vol. 1, (pp. 1–27).

Melin, H., Sandell, A., & Ihse, M. (2001b). Ctt-bank: A speech controlled telephone banking system–an initial evaluation. *TMH-QPSR*, *1*, 1–27.

Meng, H., Wai, C., & Pieraccini, R. (2003). The use of belief networks for mixed-initiative dialog modeling. *IEEE Transactions on Speech and Audio Processing*, *11*(6), 757–773.

Mentrup, C. (2004). X-motion project. Tech. rep., T-SYSTEMS NOVA Gmbh.
URL `http://cordis.europa.eu/fetch?CALLER=PROJ_ICT&ACTION=D&DOC=8&CAT=PROJ&QUERY=01325b1ac0ec:2d5e:20a955d0&RCN=63265`

Meyer, S., & Rakotonirainy, A. (2003). A survey of research on context-aware homes. In *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21*, (pp. 159–168). Australian Computer Society, Inc.

Mikulecky, P., & Olsevicova, K. (2005). University education as an ambient intelligence scenario. In *Proceedings of 4th European Conference on E-learning*, (pp. 333–341).

Minker, W., Waibel, A., & Mariani, J. (1999). *Stochastically-based semantic analysis*. Springer Netherlands.

Montoro, G., Alamán, X., & Haya, P. (2004). A plug and play spoken dialogue interface for smart environments. *Computational Linguistics and Intelligent Text Processing*, (pp. 360–370).

Muñoz, M., Gonzalez, V., Rodriguez, M., & Favela, J. (2003). Supporting context-aware collaboration in a hospital: an ethnographic informed design. In *Proceedings of Workshop on Artificial Intelligence, Information Access, and Mobile Computing 9th International Workshop on Groupware, CRIWG 2003*, (pp. 330–334). Grenoble, France.

Mylopoulos, J., Kolp, M., & Giorgini, P. (2002). Agent-oriented software development. *Methods and Applications of Artificial Intelligence*, (pp. 748–748).

Nazari Shirehjini, A., & Klar, F. (2005). 3DSim: Rapid prototyping ambient intelligence. In *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, (pp. 303–307).

Networks, N. (2011).
URL `http://www.newburynetworks.com/`

Nieto-Carvajal, I., BotÍa, J. A., Ruiz, P. M., & Gómez-Skarmeta, A. F. (2004). Implementation and evaluation of a location-aware wireless multi-agent system. In *International Conference on Embedded and Ubiquitous Computing (EUC 2004)*, (pp. 528–537).

Norman, D. (1988). *The psychology of everyday things*. Basic Books.

Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R., & Musen, M. (2001). Creating semantic web contents with Protege-2000. *IEEE Intelligent Systems*, *16*(2), 60–71.

Nwana, H. S., Ndumu, D. T., Lee, L. C., Collis, J. C., & Re, I. I. (1999). Zeus: A tool-kit for building distributed multi-agent systems. *Applied Artifical Intelligence Journal*, *13*, 129–186.

Oh, Y., & Woo, W. (2005). User-centric integration of contexts for a unified context-aware application model. In *Joint sOc-EUSAI conference*.

O'Hare, G., O'Grady, M., Keegan, S., O'Kane, D., Tynan, R., & Marsh, D. (2004). Inteligent agile agents: Active enablers for ambient intelligence. In *ACM's Special Interest Group on Computer-Human Interaction (SIGCHI), Ambient Intelligence for Scientific Discovery (AISD)*. Vienna.

Ohkubo, M., Suzuki, K., & Kinoshita, S. (2005). Rfid privacy issues and technical challenges. *Communications of the ACM*, *48*(9).

Orr, R., & Abowd, G. (2000). The smart floor: A mechanism for natural user identification and tracking. In *Proceedings of the 2000 Conference on Human Factors in Computing Systems (CHI 2000)*, (pp. 275–276). The Hague, Netherlands: ACM.

Os, E., Boves, L., Lamel, L., & Baggia, P. (1999). Overview of the arise project. In *Proceedings of the 6th European Conference on Speech Communication and Technology*, vol. 4, (pp. 1527–1530).

Öztürk, P., & Aamodt, A. (1997). Towards a model of context for case-based diagnostic problem solving. In *Context-97; Proceedings of the interdisciplinary conference on modeling and using context*, (pp. 198–208).

Paek, T., & Horvitz, E. (2000). Conversation as action under uncertainty. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, vol. 455464.

Parasuraman, A., Zeithaml, V., & Berry, L. (2002). Servqual: a multiple-item scale for measuring consumer perceptions of service quality. *Retailing: critical concepts*, *64*(1), 140.

Pascoe, J. (1997). The stick-e note architecture: Extending the interface beyond the user. In *Proceedings of International Conference on Intelligent User Interfaces*, (pp. 261–264).

Pavon, J., Gomez-Sanz, J., & Fuentes, R. (2005). The INGENIAS methodology and tools. *Agent-oriented methodologies*, (pp. 236–276).

Philips Research (2010). Myheart project webpage.
URL `http://www.research.philips.com/technologies/heartcycle/myheart-gen.html/`

Pires, L., van Sinderen, M., Munthe-Kaas, E., Prokaev, S., Hutschemaekers, M., & Plas, D. (2005). Techniques for describing and manipulating context information. *Freeband/A_MUSE/ D*, *3*.

Pokahr, A., Braubach, L., & Lamersdorf, W. (2003). Jadex: Implementing a bdi-infrastructure for jade agents. *EXP - in search of innovation (Special Issue on JADE)*, *3*(3), 76–85.

Price, P., Hirschman, L., Shriberg, E., & Wade, E. (1992). Subject-based evaluation measures for interactive spoken language systems. In *Proceedings of the workshop on Speech and Natural Language*, (pp. 34–39).

Priyantha, N., Chakraborty, A., & Balakrishnan, H. (2000a). The cricket location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, (pp. 32–43). ACM Press.

Priyantha, N., Chakraborty, A., & Balakrishnan, H. (2000b). The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, (pp. 32–43). Boston, MA.: ACM, ACM Press.

Project, T. P. (2000). Protégé.
URL `http://protege.stanford.edu`

Puerta, A. (1991). *A multiple-method knowledge-acquisition shell for the automatic generation of knowledge-acquisition tools*. Knowledge Systems Laboratory, Medical Computer Science, Stanford University.

Raab, F., Blood, E., Steiner, T., & Jones, H. (1979). Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace and Electronic Systems*, *15*(5), 709–717.

Ramanujam, S., & Capretz, M. A. M. (2004). Design of a multi-agent system for autonomous database administration. *International Journal of Intelligent Systems*, (pp. 1167–1170).

Rao, A. (1996). AgentSpeak (L): BDI agents speak out in a logical computable language. *Agents Breaking Away*, (pp. 42–55).

Rao, A., & Georgeff, M. (1995a). BDI agents: From theory to practice. In *Proceedings of the first international conference on multi-agent systems (ICMAS-95)*, (pp. 312–319).

Rao, A. S., & Georgeff, M. P. (1995b). Bdi agents: From theory to practice. In *ICMAS*, (pp. 312–319).

Rasch, K., Li, F., Sehic, S., Ayani, R., & Dustdar, S. (2011). Context-driven personalized service discovery in pervasive environments. *World Wide Web*, (pp. 1–25).

Rentto, K., Korhonen, I., Vaatanen, A., Pekkarinen, L., Tuomisto, T., Cluitmans, L., & Lappalainen, R. (2003). In *First European Symposium on Ambient Intelligence 2003, Veldhoven, The Netherlands*.

Reuss, E., Menozzi, M., Buchi, M., Koller, J., & Krueger, H. (2004). Information access at the point of care: what can we learn for designing a mobile cpr system? *International Journal of Medical Informatics*, *73*(4), 363–369.

Roy, N., Pineau, J., & Thrun, S. (2000). Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, (pp. 93–100).

Russell, D. M., Streitz, N. A., & Winograd, T. (2005). Building disappearing computers. *Communications of the ACM*, *48*(3), 42–48.

Ryan, N. (1999). Contextml: Exchanging contextual information between a mobile client and the fieldnote server. *Retrieved October*, *10*.

Ryan, N., Pascoe, J., & Morse, D. (1998). *Enhanced Reality Fieldwork: the Context-Aware Archaeological Assistant*.

S-H. Baek, J.-D. H., E-C. Choi (2007). Design of information management model for sensor based context-aware service in ubiquitous home. In *Int. Conf. on Convergence Information Technology, Gyeongju, Republic of Korea*, (pp. 1040–1047).

Salber, D., & Coutaz, J. (1993). Applying the wizard of oz technique to the study of multimodal systems. *Human-Computer Interaction*, (pp. 219–230).

Sama Rojo, V., Ferreiros, J., San Segundo Hernández, R., Pardo, J., & Fernández Martínez, F. (2005). Utilización de medidas de confianza en sistemas de comprensión del habla. *Procesamiento del lenguaje natural*, *35*(229-234).

Samulowitz, M., Michahelles, F., & Linnhoff-Popien, C. (2002). Capeus: An architecture for context-aware selection and execution of services. *New developments in distributed applications and interoperable systems*, (pp. 23–29).

Sánchez-Pi, N., Carbó, J., & Molina, J. (2008a). Analysis and design of a multi-agent system using gaia methodology in an airport case of use. In *IBERAGENTS 2008. 7th Ibero-American Workshop in Multi-Agent Systems. In Conjunction with IBERAMIA'08*.

Sánchez-Pi, N., Carbó, J., & Molina, J. (2008b). Jade/leap agents in an aml domain. vol. 5271 of *LNAI*, (pp. 62–69). Springer-Verlag.

Sánchez-Pi, N., Fuentes, V., Carbó, J., & Molina, J. M. (2007). Knowledge-based system to define context in commercial applications. In *Proc. of the 8th ACIS Conference SNPD'07*, (pp. 694–699). Tsingtao, China.

Sanchez-Pi, N., Mangina, E., Carbo, J., & Molina, J. (2011). A utility-based adaptation approach for an ami (ambient intelligence) environment. *International Journal of Artificial Intelligence*, *6*(S11), 100–111.
URL http://www.ceserp.com/cp-jour/index.php?journal=ijai&page=article&op=view&path%5B%5D=938&path%5B%5D=386

Sánchez-Pi, N., Mangina, E., Carbó, J., & Molina, J. M. (2010). Multi-agent system (MAS) applications in ambient intelligence (Ami) environments. In Y. Demazeau, F. Dignum, J. M. Corchado, J. Bajo Pérez, R. Corchuelo, E. Corchado, F. Fernández-Riverola, V. J. Julián, P. Pawlewski, & A. Campbell (Eds.) *Trends in Practical Applications of Agents and Multiagent Systems*, Advances in Intelligent and Soft Computing, (pp. 493–500). Salamanca (Spain): Springer-Verlag.

Sánchez-Pi, N., & Molina, J. (2009a). A centralized approach to an ambient assisted living application: An intelligent home. In *LNCS- Distributed Computing, Artifficial INtelligence, Bioinformatics, Soft Computing and Ambient assisted Living*. Springer Berlin/Heidelberg.

Sánchez-Pi, N., & Molina, J. (2009b). A multi-agent platform for the provisioning of u-commerce services. In *28th North American Fuzzy Information Processing Society Anual Conference*.

Sánchez-Pi, N., & Molina, J. (2010a). Adaptation of an evaluation system for e-health environments. In *LNAI-KES 2010, PART IV*, vol. 6279, (pp. 357–364). Springer Berlin/Heidelberg.

Sánchez-Pi, N., & Molina, J. (2010b). A multi-agent approach for the provisioning of e-services in u-commerce environment. *Internet Research ISSN: 1066-2243*, *20*(3).

Sánchez-Pi, N., & Molina, J. M. (2009). *A Smart Solution for Elders in Ambient Assisted Living*, chap. LNCS 5602 Bioinspired Applications in Artificial and Natural Computation., (pp. 95–103). Springer Lecture Notes in Computer Science. Springer Berlin/Heidelberg.

Sánchez-Pi, N., & Molina, J. M. (2010). *Adaptation of an Evaluation System for e-Health Environments*, chap. KES 2010, (pp. 357–364). Lecture Notes in Artificial Intelligence. Wales. UK: Springer.

Schilit, B., Adams, N., & Want, R. (1994). Context-aware computing applications. In *Proceedings of the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA,*, (pp. 85–90). Santa Cruz, CA.

Schmidt, A. (2002). *Ubiquitous Computing- Computing in Context*. Ph.D. thesis.

Schmidt, A. (2005). Knowledge maturing and the continuity of context as a unifying concept for knowledge management and e-learning. In *Proceedings of I-Know*, vol. 5.

Schmidt, A., Aidoo, A., K., Takaluoma, A., Tuomela, U., Van Laerhoven, K., & Van de Velde, W. (1999a). *Advanced Interaction in Context in Handheld and Ubiquitous Computing*, (pp. 89–101). Springer-Verlag.

Schmidt, A., Aidoo, K., Takaluoma, A., Tuomela, U., Van Laerhoven, K., & Van de Velde, W. (1999b). Advanced interaction in context. In *HandHeld and Ubiquitous Computing*, (pp. 89–101).

Schmidt, A., Beigl, M., & Gellersen, H. (1999c). There is more to context than location. *Computers and Graphics*, *23*(6), 893–901.

Schmidt, A., & Van Laerhoven, K. (2001). How to build smart appliances? *Personal Communications, IEEE*, *8*(4), 66–71.

Scholtz, J. (2001). Ubiquitous computing goes mobile. *ACM SIGMOBILE Mobile Computing and Communications Review*, *5*(3), 32–38.

Scholtz, J., & Consolvo, S. (2004). Toward a framework for evaluating ubiquitous computing applications. *IEEE Pervasive Computing*, *3*(2).

Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R., Shadbolt, N., Van de Velde, W., & Wielinga, W. (1999). Knowledge engineering and management, the commonkads methodology. The MIT Press.

Searle, J. (1969). Speech acts. *Cambridge University Press*.

Segarra, E., Sanchis, E., Galiano, M., García, F., & Hurtado, L. (2001). Extracting semantic information through automatic learning techniques. *Pattern Recognition and Image Analysis*, (p. 177).

Shadbolt, N., Motta, E., & Rouge, A. (1993). Constructing knowledge-based systems. *IEEE Software*, *10*(6), 34–38.

Sheng, Q., & Benatallah, B. (2005). Contextuml: a uml-based modeling language for model-driven development of context-aware web services. In *International Conference on Mobile Business, 2005. ICMB 2005.*, (pp. 206–212).

Shoham, Y. (1990). Nonmonotonic reasoning and causation. *Cognitive Science*, *14*(2), 213–252.

Shoham, Y. (1993). Agent-oriented programming. *Artificial intelligence*, *60*(1), 51–92.

Singh, S., Kearns, M., Litman, D., & Walker, M. (1999). Reinforcement learning for spoken dialogue systems. In *Proc. NIPS99*.

Smailagic, A., Siewiorek, D., Anhalt, J., Gemperle, F., Salber, D., Weber, S., Beck, J., & Jennings, J. (2001). Towards context aware computing: experiences and lessons. *IEEE Journal on Intelligent Systems*, *16*(3), 38–46.

Stone, P., & Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, *8*(3), 345–383.

Strang, T., & Linnhoff-Popien, C. (2003). Service interoperability on context level in ubiquitous computing environments. In *Intl. Conf. on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet*.

Strang, T., & Linnhoff-Popien, C. (2004). A context modeling survey. In *Workshop on advanced context modelling, reasoning and management as part of UbiComp*, (pp. 1–8).

Strang, T., Linnhoff-Popien, C., & Frank, K. (2003a). Applications of a context ontology language. In *Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCom2003)*, vol. 14, (p. 18).

Strang, T., Linnhoff-Popien, C., & Frank, K. (2003b). Cool: A context ontology language to enable contextual interoperability. In *Distributed Applications and Interoperable Systems*, (pp. 236–247).

Strang, T., Linnhoff-Popien, C., & Roeckl, M. (2003c). Highlevel service handover through a contextual framework. In *Proceedings of 8th International Workshop on Mobile Multimedia Communications (MoMuC2003)*, (pp. 405–410).

Sturm, J., Den Os, E., & Boves, L. (1999). Issues in spoken dialogue systems: Experiences with the Dutch ARISE system. In *Proc. ESCA workshop on Interactive Dialogue in Multi-Modal Systems*.

Sumi, Y., Etani, T., Fels, S., Simonet, N., Kobayashi, K., & Mase, K. (1998). C-map: Building a context-aware mobile assistant for exhibition tours. In *Community Computing and Support Systems, Social Interaction in Networked Communities*, (pp. 137–154). Springer-Verlag, London, UK.

Tiba, F., & Capretz, M. (2006). An overview of the analysis and design of sigma: Supervisory intelligent multi-agent system architecture. *Information and Communication Technologies, 2006. ICTTA '06. 2nd*, *2*, 3052–3057.

TILAB (2010). Jade.
URL http://sharon.cselt.it/projects/jade/

Torres, F., Hurtado, L., García, F., Sanchis, E., & Segarra, E. (2005). Error handling in a stochastic dialog system through confidence measures. *Speech Communication*, *45*(3), 211–229.

Torres, F., Sanchis, E., & Segarra, E. (2003). Development of a stochastic dialog manager driven by semantics. In *Eighth European Conference on Speech Communication and Technology*.

Traum, D., & Larsson, S. (2003). *The Information State Approach to Dialogue Management*. Current and New Directions in Discourse and Dialogue. Kluwer Academic Publishers.

Tu, S., Eriksson, H., Gennari, J., Shahar, Y., & Musen, M. (1995). *Ontology-based configuration of problem-solving methods and generation of knowledge-acquisition tools: Application of PROTÉGÉ-II to protocol-based decision support*. vol. 7. Elsevier.

United Nations (last access: Saturday, February 28, 2009; 12:01:46 AM.). World population prospects: The 2006 revision and world urbanization prospects: The 2005 revision. Tech. rep., Population Division of the Department of Economic and Social Affairs of the United Nations Secretariat.
URL http://esa.un.org/unpp

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *The Knowledge Engineering Review*, *11*(02), 93–136.

Van Halteren, A., Bults, R., Wac, K., Konstantas, D., Widya, I., Dokovsky, N., Koprinkov, G., Jones, V., & Herzog, R. (2004). Mobile patient monitoring: The mobihealth system. *The Journal on Information Technology in Healthcare*, *2*(5), 365–373.

Van Laerhoven, K. (1999). On-line adaptive context awareness starting from low-level sensors.

Vaquero, C., Saz, O., Lleida, E., Marcos, J., Canalís, C., & de Educación, C. (2006). Vocaliza: An application for computer-aided speech therapy in spanish language. *IV jornadas en tecnologías del habla, Zaragoza, España*, (pp. 321–326).

Vildjiounaite, E., & Kallio, S. (2007). A layered approach to context-dependent user modelling. *Advances in Information Retrieval*, (pp. 749–752).

Wahlster, W. (2006). *SmartKom: Foundations of Multimodal Dialogue Systems.*. Springer.

Walker, M., Fromer, J., Fabbrizio, G., Mestel, C., & Hindle, D. (1998). What can I say? Evaluating a spoken language interface to Email. In *Proc. of ACM CHI 98 Conference on Human Factors in Computing Systems*, (pp. 582–589). Los Angeles, USA.

Walker, M., Passonneau, R., & Boland, J. (2001). Quantitative and qualitative evaluation of darpa communicator spoken dialogue systems. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, (pp. 515–522).

Wang, X., Da Qing Zhang, T., & Pung, H. (2004). Ontology based context modeling and reasoning using owl. In *Workshop Proceedings of the 2nd IEEE Conference on Pervasive Computing and Communications (PerCom2004)*, (pp. 18–22). IEEE Computer Society.

Want, R., Hopper, A., Falcão, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems*, *10*(1), 91–102.

Want, R., Schilit, B., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J., & Weiser, M. (1996). The parctab ubiquitous computing experiment. In T. Imielinski, & H. F. Korth (Eds.) *Mobile Computing, chapter 2*, (pp. 45–101). Springer.

WAP Forum (2007). User agent profile (UAProf).
URL http://www.openmobilealliance.org/Technical/release_program/uap_v2_0.aspx

Watson, R. T., Pitt, L. F., Berthon, P., & Zinkhan, G. M. (2002). U-commerce: Expanding the universe of marketing. *Journal of the Academy of Marketing Science*, *30*(4), 333.

Weiser, M. (1991). The computer of the 21st century. *Scientific American*, *265*(3), 66–75.

Wielinga, B. J., From, C., Akkermans, H., Hassan, H., Olsson, O., Schreiber, G., & Terpstra, P. (1994). Expertise model definition document. technical report espirit project p5248/kads-ii/m2/uva/026/5.0. Tech. rep., University of Amsterdam and Free University of Brussels ans Netherlans Energy Research Centre ECN.

Williams, J. D., & Young, S. (2007). Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language Language*, *21*(2), 393–422.

Wooldridge, M. (1997). Agent-based software engineering. In *IEEE Proceedings on Software Engineering*, vol. 144, (pp. 26–37).

Wooldridge, M. (2000). *Reasoning about Rational Agents*. MIT Press.

Wooldridge, M., & Ciancarini, P. (2001). Agent oriented software engineering: The state of the art. In *First international Workshop on Agent-Oriented Software Engineering*. Springer.

Wooldridge, M., & Jennings, N. (1995). Agent theories, architectures and languages: a survey. In *Lecture Notes in Artificial Intelligence*, vol. 890, (pp. 1–39).

Wooldridge, M., & Jennings, N. (1996). Towards a theory of collective problem solving. *Distributed Software Agents and Applications*, *1069*(40–53).

Wooldridge, M., Jennings, N., & Kinny, D. (2000). The gaia methodology for agent-oriented analysis and design. *Journal of Autonomous Agents and Multi-Agent Systems*, *3*(3), 285–312.

World Wide Web Consortium (W3C) (2007a). Composite capabilities / preferences profile (CC/PP).
URL http://www.w3.org/Mobile/CCPP

World Wide Web Consortium (W3C) (2007b). Voice extensible markup language (voicexml). URL `http://www.w3.org/TR/voicexml21/`

Wyatt, J., & Sullivan, F. (2005). ehealth and the future: promise or peril? *British Medical Journal*, *331*(7529), 1391.

Yamazaki, T. (2005). Ubiquitous home: real-life testbed for home context-aware service. In *Testbeds and Research Infrastructures for the Development of Networks and Communities, 2005. Tridentcom 2005. First International Conference on*, (pp. 54–59).

Yang, S. J. H. (2006). Context aware ubiquitous learning environments for peer-to-peer collaborative learning. *Educational Technology and Society*, *9*(1), 188–201.

Yaua, S. S., & Karim, F. (2004). A context sensitive middleware for dynamic integration of mobile devices with network infrastructures. *Journal of Parallel and Distributed Computing*, (pp. 301–317).

Yoo, B., & Donthu, N. (2001). Developing a scale to measure the perceived quality of an internet shopping site (sitequal). *Quarterly Journal of Electronic Commerce*, *2*, 31–46.

Young, S. (2002). The Statistical Approach to the Design of Spoken Dialogue Systems. Tech. rep.

Young, S., Schatzmann, J., Weilhammer, K., & Ye, H. (2005). The hidden information state approach to dialogue management. In *Proceedings of ICASSP*.

Young, S., Schatzmann, J., Weilhammer, K., & Ye, H. (2007). The hidden information state approach to dialog management. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, (pp. iv–149).

Zelkha, E., & Epstein, B. (1998). From devices to ambient intelligence. In *Digital Living Room Conference*.

Zhang, X., & Prybutok, V. (2005). A consumer perspective of e-service quality. *IEEE Transactions on Engineering Management*, *52*(4), 461–477.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., & Hetherington, L. (2000a). JUPITER: A telephone-based conversational interface for weather information. In *IEEE Transactions on Speech and Audio Processing*, vol. 8, (pp. 85–96).

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., & Hetherington, L. (2000b). JUPITER: A telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, *8*(1), 85–96.