Journal of Universal Computer Science, vol. 16, no. 12 (2010), 1500-1520
submitted: 29/12/09, accepted: 5/5/10, appeared: 28/6/10 © J.UCS

An Agent-based Architecture for Developing Activity-Aware Systems for Assisting Elderly

Juan Pablo García-Vázquez[1], Marcela D. Rodríguez[1], Mónica E. Tentori[2]
Diana Saldaña[1], Ángel G. Andrade[1], Adán N. Espinoza[1]
([1]MyDCI Program of the Engineering School, UABC, Mexicali, Mexico
[2] School of Computer Sciences, UABC, Ensenada, Mexico
{pablo.garcia, marcerod, aandrade, nespinoza}@uabc.edu.mx; {mtentori, dianasj}@uabc.mx)

Abstract: Ageing is a global phenomenon which has motivated many research and development projects with the aim of providing computing services that support the active and independent living of the elderly. To integrate the ambient intelligence (AmI) vision into the home environment to allow elders to "age in place", it has been identified the necessity of providing high-level software support for creating ambient assisted living (AAL) environments. We propose activity-aware computing to allow smart environments to provide continuous activity awareness and opportunistically offer assistance aimed at supporting the elders' current activity. This new paradigm calls for novel tools to help developers mirror human activities in the digital domain, and adapt smart environments based on the activities executed by the users. This paper proposes the use of autonomous agents to cope with the design issues for developing activity-aware systems. We specialized the SALSA agent architecture by incorporating customizable activity-aware mechanisms to infer and represent activities. We illustrate the capabilities offered by SALSA autonomous agents through a design of an activity-aware application for helping elders to manage their medication activity.

Keywords: Multiagents systems, artificial intelligence, smart environments, knowledge representation formalisms and methods, Information Interfaces and Presentation, Ubiquitous Computing
Categories: I.2.6, I.2.11, I.2.4, H.5, L.7

1 Introduction

The ubiquitous computing research community has identified the necessity of providing high-level software support for creating ubiquitous computing systems for specific application domains [Gu, 05]. In this sense, some middleware's have been proposed for domains such as the healthcare [Bardram, 07] and ambient assisted living (AAL) environments for supporting "ageing in place" [Keng, 09]. Ageing is a global phenomenon which has motivated many research and development projects with the aim of providing computing services that support the active and independent living of the elderly. A current technological trend is to integrate ambient intelligence (AmI) vision [Bravo, 06] into the home environment to allow "Recent Adults" who do not depend on others to carry out their basic and instrumental activities, to "age in place" [Zao, 08][Horgas, 04]. However, we need to consider in the system design, that these elders may present a cognitive decline that naturally they get due ageing. Therefore, they may face some problems for carrying out these activities at their

homes, such as forgetting some task or step for performing an activity. An AmI home-environment that assists elderly should include not only the pervasive technology to track their behaviour and interaction with household objects, but the context-aware and intelligent systems to infer and monitor activities of daily living (ADLs) and to recognize when elders needs to be assisted [Cook, 06][Kimel, 07]. Context-awareness will allow such an environment to take on the responsibility of serving users by tailoring itself to their particular needs and skills, and to perform tasks according to the nature of the physical space.

Activity-aware computing is an appropriate approach for implementing AmI systems for elders, since it defines the principles that enable developers to understand how users' activities can be supported with ubiquitous computing technology [Tentori,08] [Bardram,07]. Activity-aware technologies are defined as tools that allow AmI environments to automatically infer activities and opportunistically offer services that support the user's task at hand without intruding in their focal activity but acting based on it [Tentori,10]. With such applications, health and behavioural data can be captured, analyzed, and mined over time providing valuable evidence for care delivery. Activity-aware applications need intelligent capabilities to be adaptive and reactive to users' tasks and learn from users' behaviour to provide high quality services. AI techniques such as autonomous agents, models for context representation and techniques for context recognition can provide developers with mechanisms that take advantage of the user's context to infer and represent activities.

Developing a middleware is a critical factor in moving activity-aware applications beyond research prototypes. The aim of this paper is to present how SALSA, an infrastructure for creating agent-based ubiquitous computing systems [Rodriguez, 05], was specialized to address complexities such as the aforementioned in order to facilitate the development of activity-aware applications that assist elders in performing their activities of daily living. We extended the SALSA architecture by creating services and components that allow autonomous agents to model computational activities, which can move across the devices used by elders in order to provide them with timely aids. Through a set of specialized SALSA communication messages, agents can announce activity events and gather context for inferring high-level context, such as activity recognition and risks associated with an activity.

Before presenting the specialized services and API of the SALSA architecture, Section 2 presents work related to architectures for AmI systems. Section 3 presents the results of a case study which helps us to understand the elders' needs for medicating, and to envision scenarios of activity-aware applications that support them in this activity. Section 4 describes the design issues to be addressed through autonomous agents which enabled us to determine how the SALSA middleware should be extended, as it is presented in section 5. Section 6 illustrates how an activity-aware application can be easily implemented with the extended SALSA architecture. And finally, section 7 presents the conclusions and future work.

2 Related Work

2.1 Agent-based architectures for AmI environments

Several architectures have been proposed for addressing different challenges for developing ambient intelligence environments by using agents. The LAICA project aims to provide a distributed middleware that addresses the performance of AmI systems, including issues of synchronization and coordination among agents by providing a look up table of the available agents in the environment and an intelligent component that decides when agents should provide their services[Cabri, 05]. Thus, this middleware does not provide an API for creating AmI systems, but a set of components that enables the coordination of AmI agents. Other middlewares provide facilities for capturing context information, such as the middleware presented in [Soldatos, 07], which allows flexible addition and replacement of hardware that captures context information (e.g., sensors, actuators). It is composed of a set of agents that model and track higher-level contextual situations by using ontology-based directory services which enable the inference of higher-level context information based on existing meta-data. In this direction, other frameworks incorporate mechanisms to model context by introducing two-layered ontologies and providing efficient support for acquiring, discovering, interpreting and accessing various contexts [Gu, 05] [Bochini,07]. To use such context models a developer must create instances of it stipulating the type of context he is modelling and the rules associated with it. In contrast, in our approach the middleware is able to automatically hypothesize context in the form of activities. Finally, agent-based architectures developed for addressing the challenges of AmI environments for specific domains have been proposed. For instance, the GerAmi system is an agent-based architecture to implement a distributed and intelligent environment to help healthcare providers contend with the challenges of caring for Alzheimer's patients, the elderly, and people with other disabilities. The GerAmi agents were implemented by using the deliberative (BDI) architecture implemented with a Case-Based Planning system, which improves the agents learning capabilities. And consequently, GerAmi agents increased the time spent by medical staff for direct patient care, and reduced the time spent attending to false alarms, supervision and control-tasks [Corchado, 2008]. Even though some of the abovementioned architectures considered in their design a way to provide a model for representing context, including users' activities and tasks, they did not consider that the applications developed on top of them should provide activity awareness, adaptation and recognition.

2.2 Architectures for activity-aware computing systems

We have identified limited work for proposing middlewares that address the complexities for creating activity-aware computing. The ABC framework provides the design principles and technological support for developing activity-based computing systems. It provides components that manage communication for activity and state management, and to handle the activity storage, roaming, and sharing (which were design principles identified by studying the activities in a hospital setting). For modeling the activity state, it uses a generic XML activity ontology called Activity Markup Language (AML) [Bardram,07]. Other research efforts are

more oriented towards proposing high-level guidelines for developing activity-aware applications with an emphasis on implementing systems for the recognition of activities [Mahmud, 09]. Finally, several projects have explored algorithms, methods and devices for the automatic recognition of users' activities. For example, the PROACT toolkit was proposed to improve HCI techniques for recognizing activities of daily living (ADL) by using RFID technology combined with the Monte-Carlo inference engine [Philipose, 04]. Thus, the lack of general frameworks for creating activity-based applications motivated us to specialize the SALSA middleware by incorporating in it agent-based activity-aware services and by extending its API for enabling activity-awareness among the system's agents. To identify the SALSA requirements to implement activity-aware applications, we analyzed the context of use of specific domain applications, such as hospital settings [Tentori, 10] and, the elders' activities of daily living performed at their homes as we report in this paper.

3 Characterizing Elders' Activities of Daily Living

To identify the kind of support that the elderly may need for performing their ADLs in their homes, we carried out a case study by following the activity-centered design approach [Nardi,96], wich allow us to understand and model how elders perform their activities, to identify the objects and/or tools they use to perform them and the community that participates. Specifically, we carried out a case study to understand the elders' needs and problems for managing their medication. It consisted of two stages. The first one included 20-minute semi-structured interviews based on medical instruments for measuring the elders' functional capability to perform their basic and instrumental activities [Lawton, 90]. From 28 elders ranging in age from 63 to 86 years old, we selected 17 whom according to these instruments were able to live independently in their homes. The second stage included 40-minute semi-structured and contextual interviews based on the MedMaIDE index [Orwing, 96]. This is an instrument to determine the deficiencies of elders when attempting to adhere to their medication prescription, since it has been identified that persons who are over 65 years are the most likely segment of the population to face frequent problems associated with non-adherence. The study case results show that elders are aware that they face problems for adhering to their medication prescription. Therefore, they have created their own strategies for coping with these problems as we explain in the following sections.

3.1 Strategies for medication adherence

Elders personalize and adapt their medication routines by following strategies when managing their medication:

- *Have a specific place to medicate.* We found out that most of them medicate in the kitchen (11/17), or in a room near the kitchen such as the living room (2/17), and a few (4/17) medicate in the bedroom. Elders elected to medicate in those places where they spend more time to facilitate remembering when they need to take their medication or have the tools they need at hand. For instance, the kitchen is the most used place for medicating, since it is the place in which they find the tools they use to medicate (i.e. glass of water).

- *Order and store their medicines in specific containers.* As participants are afraid of under or over medicating, they (6/17) use containers or select a specific place in the home (such as a drawer in the kitchen cabinet) to store their medicines and order them according to different criteria, such as the medicine doses.
- *Maintain notes indicating the purpose of taking their medicines.* Some elders (3/17) stated that they have notes that they or their doctor write down specifying the health purpose of taking each medicine.
- *Visit their doctor periodically to refill their medicines.* All older adults indicated that they periodically visit their doctor and refill their medicines from the clinic's pharmacy after seeing the doctor who gives them a new prescription. These older adults use a calendar or reminders in the form of notes to remember the date of the appointment with the doctor. Additionally, (11/17) they require help from a family member who takes them to the clinic and pharmacy.

3.2 Modelling the elders' management of medication

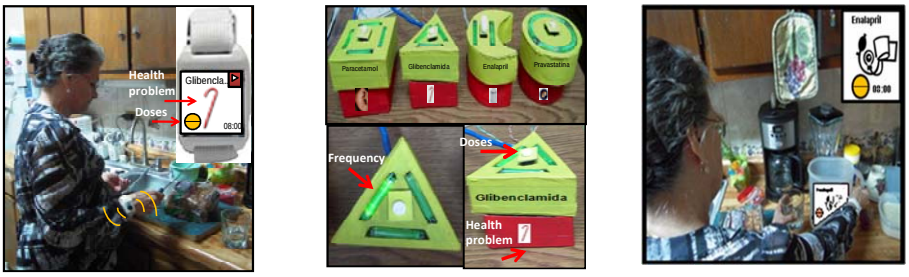
Modeling elders' medication management routines enabled us to identify and represent how elders carry out their actions through supporting resources that help them to accomplish their medication goals, such as the community that helps them to refill their medications, and the objects or tools used to take their medicines. We identify the sequencce of actions followed by elders when it is time for medicating. Elders move to the place in which they usually medicate, such as the kitchen; then they start to perform different actions by interacting with the objects or tools they need to medicate such as, a glass of water (basic object) or a specific medicine container (complementary object). We considered an object or tool as basic if elders use it for medicating by following the physician's instructions; in contrast, we considered an object as complementary when they show a preference for using it to facilitate the medication process. Complementary objects are part of elders' strategies followed to personalize and adapt their medication routines according to their skills and abilities. We also identified the actions followed by the elders that need the help of their children for visiting their doctor periodically in order to refill their medication. In this case, we show that the elder uses a medical card in which the hospital nurse writes down the next appointment date. The elder may additionally use a personal calendar (complementary object) to write down the appointment in order to remember it. The characterization of the medication routines helped us to envision scenarios that show how activity-aware computing could augment the elders' home environments in two ways: (1) by assisting them to complete a medication routine by providing activity awareness and timely notifications in an expressive manner and (2) by providing elders with activity awareness to prevent potential risks associated with a specific routine, i.e. to forget going to the physician's appointment which may impede they acquire their medicines. Elders' require technological support to complete their activities of daily living free of risk. There are several challenges in designing this kind of technologies since we have to consider that older adults are characterized by presenting a natural cognitive decline, a decreased in their attention levels, and that they may consider that new technologies are complex to use. Elders need systems that provide them with awareness of their activities, and that adapt to the aforementioned characteristics. In the following scenarios, we illustrate how

activity-aware computing systems facilitates to provide this activity-awareness and adaptation to promote elders autonomy.

3.3 Scenarios

3.3.1 Activity-aware ambient displays for assisting elders to take their medications

Mrs. Maria is 72 years old. It is noon and while she is preparing her meals, her wearable notification display generates an audio notification to make her aware that it is time to take her medicines (figure 1a). When Maria approaches her medicine dispenser the e-activity in her wearable display jumps to the dispenser to notify her that she must take two medicines (figure 1b). For each medicine, the dispenser has containers in the form of geometric figures. The number of sides of each figure indicates the number of times that she has to take a medicine per day (frequency). The dose to take is shown in the middle of the container. Thus, Maria realizes that she has to take one Glibenclamida pill, and one Enalapril. Then she takes her pills, and when she serves the water for taking them, the pitcher presents additional information to persuade her of the importance of these medicines for their health, such as that Enalapril is for controlling her blood pressure as presented in figure 1c.



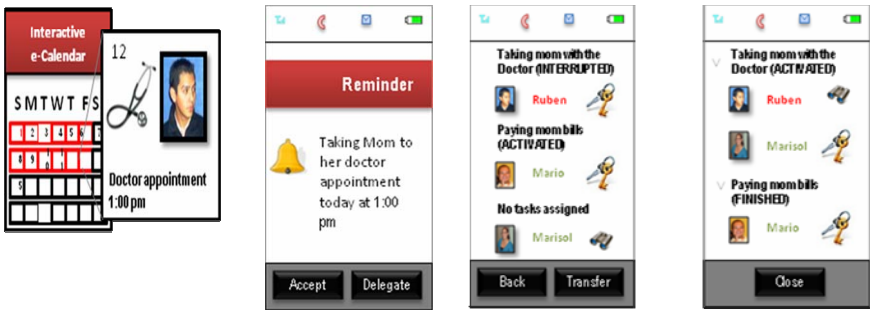
a) b) c)

Figure 1: Activity-aware system providing aids for medicating on different objects: a) Wearable Notification Display; b) Medicine Dispenser; c) Notifying aditional information of her medication.

3.3.2 Children helping parents to reach their medication adherence goal through an activity-aware system

Every month, Maria visits the doctor to be consulted and to refill her medications in the clinic's pharmacy. When Maria returns home, after being consulted by her physician, she schedules her next appointment with the doctor on her interactive e-Calendar as presented in figure 2a. Consequently, her son Ruben (who usually takes her to the clinic) receives a notification on his mobile phone as shown in figure 2b, which updates his mobile calendar. However, as Ruben did not foresee that the same date of the doctor's appointment he would need to make some unplanned work activities, he decides to transfer this routine to one of his siblings: Marisol and Mario. Ruben sees his mom's activity-aware roster and notices that Marisol is currently available since she did not have an assigned task for helping his mom, as

depicted in figure 2c, so he asks her to take their mother. Later, Ruben realizes that Marisol has accepted doing this task as presented in figure 2d. Therefore, Marisol has the owner role of the e-activity and Ruben is an observer since he is interested in receiveing notification of the activity status. The roles of Ruben and Marisol are depicted by the icons next to their names.



a) b) c) d)

Figure 2: Activity-aware system to enable elder's community to support her by: a) enabling them to coordinate, b) providing reminders, c) showing the activity was extended or d) transferred to other community member to be supported.

3.3.3 Discussion

The ambient display applications presented in the scenarios react and adapt their behavior based on individuals' executed activities. For instance, in scenario 1, while Maria moves in her environment to interact with different objects or artifacts that she uses to medicate, these objects need to get the activity state and execution context to provide appropriate awareness. For instance, when Maria goes to her medicine container, it reacts by presenting information that will assist her to medicate. Thus, we observed that the used objects, which were digitally augmented, react according to the activity execution context, such as Marias's location. In scenario 2, we observed that the elder's medication activity is extended to be supported by other members of her community when the elder needs to visit the doctor. In this case, the general goal is that "Maria Adheres to her Medication" which is shared with the members of her community that support her (her children and doctor). However, in order for the community members support the elder in her activity they have individual tasks or routines to accomplish, which are associated with specific goals. For instance, Ruben has the goal of "taking Maria to the doctor", which will enable Maria to reach the goal of "refill her medicines" to continue adhering to her medication routine. Additionally, a community member may require transferring to his assigned task another member in order to support the elder's activity, as illustrated in Scenario 2 when Ruben asks Marisol to take their mother to the doctor. We identified that activities can be the key computational units of the activity-aware applications presented in the scenarios. We defined that these key computational units are e-activities which computationally represent the human activity [Tentori,08].

4 Autonomous agents for managing e-activities

To implement smart environments taking into account e-activities as their basic computational unit, we decided to use autonomous agents as appropriate system constructs due to their characteristics. An e-activity is the computational representation of a human activity and it is defined by its attributes, its execution model and behaviour [Tentori,10]. Similar to e-activities, autonomous agents are software entities that can perceive different types of information from their environment and react accordingly, while acting autonomously and proactively.

We have identified some of the design issues for implementing activity-aware systems for supporting elders to perform their ADLs. Based on these design issues and by considering the capabilities that characterize autonomous agents, such as reactivity, mobility, and collaboration, we identified the SALSA middleware requirements to be addressed with autonomous agents. We considered that autonomous agents are abstraction tools that provide smart environments with the means to adapt their behaviour based on the elder's executed activity, resulting in an agent-based activity-aware system. We mainly identified that the design issues to address for implementing activity-aware systems for the elderly are related to the mechanisms needed for representing e-activities and for managing the e-activity behaviour and characteristics.

4.1 Representing e-activities

The attributes of an e-activity are essential characteristics that determine its uniqueness and depict its execution context. Such attributes include user and environmental contextual information and the rules that define how the environment expects users to act when some contextual conditions are met. e-Activity attributes include contextual information regarding who is performing the action, the location in which it is being executed, and the objects or artefacts used. For instance in the first scenario, when the elder Maria executes different actions for taking her medication by using different objects such as the medicine container and water pitcher, they provide relevant information like the purpose of taking her medication, the adequate medication doses and give her feedback related to those medicines that were taken.

Thus, an e-activity follows an execution model according to the way users execute their activities in real life. To enable elders to perform their ADLs independently, e-activities are aware of the users' context to autonomously activate an action plan, and then start the execution of a set of actions or operations to follow such plan. A user might perceive interruptions or changes in the environment, and then decide to suspend his activity which will cause the e-activity to move to an interrupted state. As a consequence, a user may also resume an e-activity when going back and forth from one activity to another [Bardram, 07]. Autonomous agents thus need mechanisms that facilitate creating a representational model of users' activities which support specialized queries for retrieving different modules of the model or activity histories. This representation model should have as its main elements: users' executed activity, its attributes and its execution context.

4.2 Managing e-activities behavior

An e-activity is characterized by presenting the following behaviour [Tentori,10]:

- *Reactivity*—it should perceive changes in its attributes and its execution context to enforce rules adapting its behavior. For instance, as shown in Scenario 2, when the e-activity perceives that Marisol will take Maria to the doctor instead of Ruben, it adapts the objects associated with it, such as Maria's calendar and the mobile devices of Marisol and Ruben, to provide them with awareness of the activity status.
- *Sequential*—it will be part of a history of activities, actions or operations aimed at following a user's envisioned plan. For instance, after a medicine reminder is received by Maria, she approaches the dispenser which shows her critical information that she needs to take each medicine.
- *Adaptability* —it must be able to adjust its representation by modulating the details of shared information. An example of this is in Scenario 1, when the information shown in Maria's bracelet is adapted to a light in her pills dispenser.
- *Persistence*—when an activity is suspended, it can be stored over long periods of time waiting to resume. For instance, in Scenario 1 while Maria is medicating, some food was left on the stove; thus, Maria may require suspending her medication activity to check her food, and then go back to take her medicines, which causes the e-activity to be reactivated by prompting awareness of her medication status.

4.2 Providing activity-aware aids

We found out that when an e-activity is natural and ubiquitously integrated into the environment where elders live and interact with the objects they use on a daily basis, an e-activity makes hops or is extended to aid the elderly. These two characteristics were identified from the results and scenarios obtained from the case study.

4.2.1 e-Activity hops

An e-activity must be able to migrate (move) across:

- *Devices* — An e-activity hops to the devices that the elder is using while performing his activity in order to be assisted opportunistically. To do this, an e-activity modulates its representation by taking into account the capabilities of each device that will display information based on the e-activity execution context which is determined by its attributes. For instance, as shown in Scenario 1, when Maria's e-activity of medicine administration hops from her bracelet to the pills dispenser and finally to the glass, these devices provide appropriate aids by considering the e-activities attributes such as the user's actions, used objects and location. An e-activity migration must be performed autonomously to continue its execution on a new destination. More specifically, autonomous agents could use mobile mechanisms to transport: the e-activity's attributes and rules for enabling agents to decide how to assist elders according to the device's capabilities; i.e. when Maria approaches and interacts with the pills dispenser then the e-activity hops to it and is modulated according to the dispenser capabilities for presenting assistance to her.
- *Owners* —an e-activity must hop from one community member to another to allow the transferring of the e-activity. To implement the functionality illustrated in scenario 2, the autonomous agent representing the e-activity owner needs to transfer to the agent of next e-activity owner, the e-activity attributes and the rules to follow

in order to execute the activity as planned. Owners' autonomous agents also need to be aware of the transitions of the e-activity states and the roles played by them during the activity. Thus, when Marisol agrees to take her mom to the doctor, she received the necessary information to carry it out as planned by Ruben and her mom. Additionally, Ruben will be aware when Marisol takes his mom to the doctor.

4.2.2 Extending the e-Activity to be supported

We observed that in order to reach an e-activity owner's goal, the e-activity could be extended to be supported by other persons of the elder's community. For instance, as shown in Scenario 2, Maria's children help her to reach her goal of visiting the doctor in a timely fashion in order to get a new medicine prescription to refill her medicines, this without sharing the same goal but having a different interest in the activity. For this case, Maria owns the goal of "refilling her medicines", it is not shared by Ruben but supported by him. This differs from previously defined concepts regarding sharing and transferring computational activities provided by Bardram & Christensen, 07. To implement this scenario functionality, autonomous agents of the elders' community member need to access the e-activity attributes and the rules that govern the plan to follow to execute the activity.

5 Extending the SALSA architecture

To enable developers to cope with the aforementioned requirements for managing e-activities in smart environments, we extended the Simple Agent Library for Smart Ambients (SALSA) [Rodriguez,05]. SALSA was developed to facilitate the creation of the software entities of an ambient computing environment with which users need to interact seamlessly[Rodriguez,08]. SALSA provides abstractions and services that enable the easy implementation and execution of autonomous agents that are reactive to context information, and communicate with other SALSA agents and users by using the same communication protocol and platform. In this paper, we present how SALSA enables the easy implementation of activity-aware applications by extending its communication protocol and services.

5.1 SALSA middleware

Figure 3 presents a physical view of the architecture of the SALSA middleware which consists of packages containing the classes and nodes containing the SALSA services.

The Agent Broker service handles the communication among agents representing the pervasive devices, services and users of the smart ambient. An agent interacts with other agents and users through the Broker by using a Broker Proxy, which is an agent's component. The Broker stores the state of people and agents and notifies their changes to other agents subscribed to them through XML messages. Developers can define new states according to the device or service the agent represents. For instance, an agent representing a printer may notify if it is busy or has a paper jam. The Client and Utilities packages provide the classes to facilitate the development of the agent's proxy to the Broker and to parse the XML messages communicated via the Broker.

The Agent Directory service allows agents to register their services when they are initialized, and to look for services available in the environment during execution time by using the SALSA Agent class Framework. The Agent Directory is accessible through a SALSA agent acting as proxy to the Agent Directory. The SALSA Agent class Framework package includes the collection of SALSA classes grouped in collaborations that shape the way in which they were organized. We represent in these collaborations additional information regarding the main classes that participate in them and the design patterns used to implement a desirable agent behavior. Thus, this package provides the Application Programming Interface (API) to implement the agent's components for perceiving, reasoning and acting. SALSA agents gather context information from users or other agents by using a Passive Perception which was implemented based on the Observer design pattern. This type of agent perception starts when a user or agent sends data to other agent through the Agent Broker. Agents can also perceive context directly from other hardware and software components, such as sensors or graphical user interfaces by using an Active Perception whose implementation is based on the Adapter design pattern. The perceived information generates events which are captured by the Reasoning component that governs the agent's action. The programmer, based on the logic of the agent, implements this component using a reasoning algorithm, such as a simple condition-action rule or a neural network. SALSA provides abstractions that permit easily incorporating and modifying any reasoning algorithms requiring little or no modifications to the other agent's components. The Acting component is mainly implemented by an abstract class to later enforce actions based on the agent's reasoning. It also includes the set of methods that implements the SALSA communication protocol which is an expressive language to enable the exchange of different types of objects between agents (such as perceived information, requests for services), between agents and users (such as events generated by the user's actions), and between agents and services (such as the state of a service) via the Agent Broker.

A detailed presentation of the architecture and protocol of SALSA, as well as sample applications in the domain of healthcare can be found in [Rodriguez, 05]. In the following sections, we present the extension of SALSA to facilitate developers to implement activity-aware applications for the domain of assisting elders to carry out activities of daily living. To reach this end, we added an Activity-aware Service to enable application agents to be aware of the users' activities by making specialized queries to the Knowledge Base of e-Activities and by extending the Agent Broker to handle the interaction among users and the agents that implement e-Activities.

5.2 Activity-aware Service

The Activity-aware Service manages the Knowledge Base of e-Activities which is a Context-Aware Representation of e-Activities (CARe). It contains information that shapes the way the environment expects that an older adult behaves to carry out his activities of daily living, and information regarding the history of activities carried out by him. To enable creating histories of activities we constructed a context representational model which is an ontology that provides a common language for agents to communicate the context of the users. Additionally, the CARe Knowledge Base contains rules identified by the developer as the conditions that have to be monitored to decide the type of assistance that an older adult requires.

5.2.1 CARe Ontology

Compared to the ontologies presented in the related work section, which were designed for representing the general concepts that characterize users, network, devices and applications of an ubiquitous computing environment [Gu, 05] [Bochini,07], we designed the CARe ontology to represent concepts specifically related to the activities of users. To reach this end, we carried out case studies based on the activity-centred design approach which enabled us to identify the concepts and relationships that characterize the activities of medical staff in a hospital setting [Tentori,10], and of elders carrying out their activities of daily living at their homes as it is reported in this paper. Thus, the CARe ontology is general enough for representing the context of users performing their activities, and it can be extended by developers to represent users' activities in specific domains.

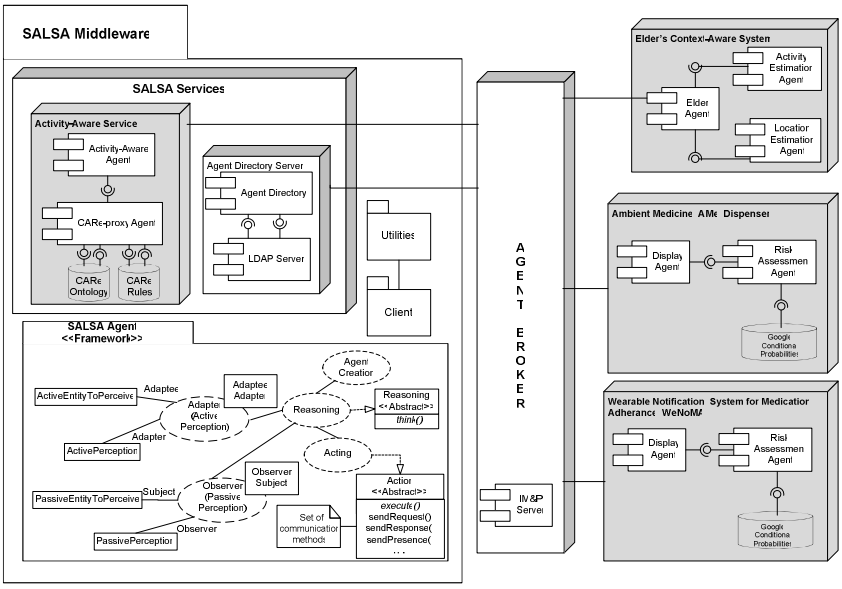


Figure 3: SALSA architecture

The design of the ontology, presented in figure 4, indicates that an Activity may be linked to other activities since they are commonly performed sequentially or in a certain order; i.e. following the physician's instructions, the elder takes her medications after each meal. An Activity is composed of Actions or sub-Activities; for example, when the elder is going to perform her medication taken routine, she first goes to her medicine dispenser, select her doses, and then takes a glass of water to take her medicine pills. Elders use objects that are indispensable for carrying out an activity routine, these objects are categorized as Basic Objects, i.e. the pills dispenser or containers, the glass of water to take the pills and the medical card used to schedule the next visit to the hospital to visit the doctor and refill medications. Monitoring the interactions with these Basic Objects helps to infer the users' activities. Additionally, users may use Complementary Objects to support or enhance the execution of an

activity, such as the elder using a bracelet to receive timely reminders for taking her medicines as illustrated in Scenario 1. From our case study we identified that elders' activities are characterized by the following Attributes: the Person who executes the activity, the StartTime and FinishTime of the activity, and the Location where the activity is being executed. An activity may have other attributes, such as the Frequency of executing it, the followed Schedule, and Community which indicates that other persons participated in the activity, for instance, the older adult's child helping her to visit the doctor. The CARe ontology can be extended by developers to add other attributes of a specific domain, i.e, a basic Activity of Daily Living (ADL) has specific properties which are the elder context that need to be monitored to infer a potential risk or abnormal situation, such as the Duration of taking a shower or a nap. Attributes are used to determine if a CARe Rule has been met.

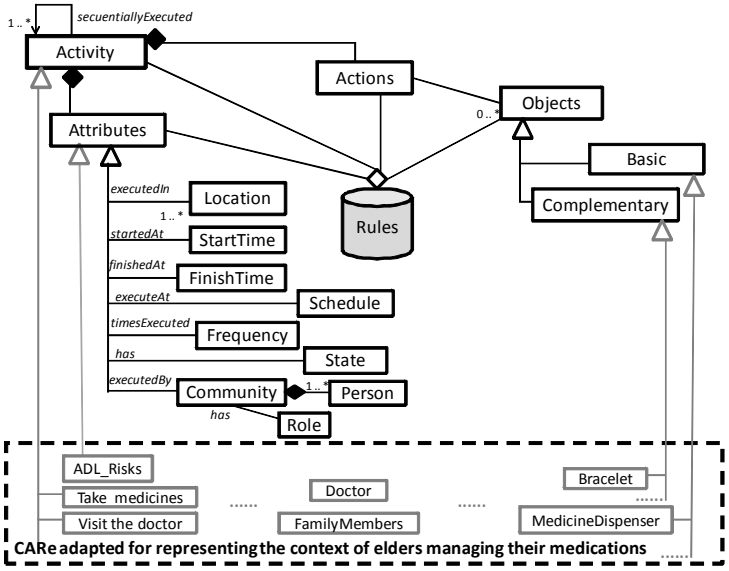


Figure 4: CARe, the Context Aware Representation of e-Activities

The CARe ontology was implemented with the OWL language, which may enable non-SALSA applications to access the CARe knowledge base to read elders' histories of activities. For instance, an Electronic Health Record System may update the elders health records based on the CARe ontology content. Additionally, this may also facilitate that developers extending the CARe ontology by connecting it with other ontologies to represent a broad range of contextual variables [Gu, 05] [Bochini,07] that include the network and devices characteristics of the AmI environment.

5.2.2 CARe Rules Base

We identified two types of rules to be contained in the CARe knowledge base:

- *Rules that govern how the environment should assist elders.* These rules enable the system agents to decide if an e-activity should hop or be extended to provide appropriate aids to the elder. Agents' decisions are made by taking into account the elder's context which is pictured in the CARe Ontology. For instance, when the elder is in front of the medicine dispenser at the time to medicate, then the e-activity has to hop to this device to be modulated in order to guide her in taking appropriate doses of each medicine. In this case, since the assertions of the left side of the rule 5 of figure 5 are met, then the action to follow is to provide the medicine dispenser with the information to be displayed, which is obtained from the rules modeling the elder expected behavior.
- *Rules modelling the elder's expected behavior*. These types of rules provide a representation of how the system expects an elder behaves according to her particular needs for carrying out her activities of daily living. Thus, for implementing the system of Scenario 1, developers created rules indicating that Maria should take one pill of Pravastatina twice at day, and one pill of Glibenclamida three times daily after each meal as presented in figure 5a (rules 1-3). While in Scenario 2, when Marisol agrees to take her mother to visit the doctor, the activity-aware system on her mobile device obtains the rules to follow in order to generate timely reminders as shown in figure 5b (rule 6).

```
1→ (defrule getPrescriptionPravastatina
(Prescription {time == morning || time ==
noon}) => ((assert location kitchen)
(assert name Pravastatina) (assert doses
1) (assert frequency 2) (assert
medicationTime time) (assert
healthProblem cholesterol)))

2→(defrule getPrescriptionGlibenclamida
(Prescription {time == morning || time ==
afternoon || time == noon}) => ((assert
location kitchen) (assert name
Glibenclamida) (assert doses 1) (assert
frequency 3) (assert medicationTime time)
(assert healthProblem diabetes)))

3→(defrule getDoctorAppointment
(Appointment { date == 2010-02-29}) =>
((assert activity  VisitingDoctor)
(assert time 1:00) (assert
doctorAppointmentDate date) (assert
communityMember Marisol@jabberserver)))
```

a)

```
4→(defrule provideReminderWeNoMA (Attributes
{ time == medicationTime}) => ((assert
agentToProvideAid WeNoMA@jabberserver )))

5→(defrule provideReminderAMe (Attributes
{time == medicationTime && location ==
InFrontTheMedicineDispenser}) => ((assert
agentToProvideAid AMe@jabberserver)))

6→(defrule provideReminderAppointment
(Attributes {date == doctorAppointmentDate})
=> ((assert agentToProvideAid
communityMember)))
```

b)

Figure 5: CARe Rule Base created with JESS. a) Rules modelling the elder's expected behaviour. b) Rules governing the activity-aware system actions to assist elders.

The content of the CARe Rule Base will be defined by developers according to the activities to be supported by the activity-aware system to implement. Having two different repositories for each type of rules, enables developers to easily adapt the activity-aware application according to the assistance to be provided and to the particular needs of the elders that will use it. That is, developers have to model the activity-aware application behavior by specifying the general rules that will govern the system agents' actions to be executed to aid any elder. Additionally, in order for the application to provide assistance that addresses the particular needs of a specific

user, developers have to update the rules that model the user's expected behavior (i.e rules specifying how Maria should take her medicines). Thus, while a set of general rules of an activity-aware application are implemented once when developing the system; the set of rules that models the elders' behaviour are implemented for each user.

5.2.3 SALSA Agents of the Activity-aware Service

The activity-aware SALSA service is responsible for controlling the lifecycle of e-activities and the interaction with the CARe Knowledge Base. The activity-aware SALSA service also allows the programmer to decide which functions are associated with each e-activity behavior: adaptability, mobility, reactivity and persistence. To enable the e-activity to hop in order to adapt services and information according to the users needs, agents retrieve specific activity histories and rules that govern the behaviour of the environment devices. The Activity-aware service of SALSA contains the Activity-aware Agent that gathers the user's context as perceived by system agents developed with specialized algorithms and/or devices, for instance, the Location-estimation agent and Activity-Estimation Agent developed by the system programmer to infer the user's location and activity respectively. Explaining the reasoning algorithms of these Agents are out of this scope of this paper, but further information can be found elsewhere [Rodriguez,05][Tentori,10]. The Activity-Aware Agent perceives the context information through a presence message sent by the agent that represents the user in the environment (i.e the Elder Agent). Thus, when the elder's context changes (such as the location or the activity), this is wrapped in an extended presence message as illustrated in Table 1. To update the CARe ontology, the Activity-aware Agent sends the perceived context information to the CARe-proxy agent. It allows other system's agents to update and retrieve the CARe ontology and rule base by using the extended SALSA API as we explain in the next section.

5.3 SALSA API for managing e-activities

To manage the execution model of e-activities, we extended the Extensible Messaging and Presence Protocol (XMPP) of the IM&P server by implementing the Agent Broker to facilitate the system agents and users to be aware of the presence and states of e-activities. These SALSA extensions enable agents and users to handle their level of awareness of e-activities through an API (methods and events classes) that facilitates composing, sending, and receiving messages between agents. Table 1 presents some of these methods, their content of the message, and the SALSA event generated by the perception component of the Agent that receives the message.

Method for requesting an action or service	**Example of an XML message sent by the SALSA method**	**Event generated when the message is perceived**
sendUpdateActivity(Activity, Person, Role)	<iq to='ContextAware-Agent@jabberserver'> <query xmlns='jabber:iq:roster'> <item jid='Maria-Agent@jabberserver' name='Maria' subscription='both'> <group>"Cooking"</group> </item> </query> </iq>	UpdatedActivityEvent
sendPresenceActivity Attributes(Activity,Ac tivityAttributes)	<presence from='Maria-Agent@jabberserver'> <x xmlns="xmpp:x:attributes"> <location>NearTheStove </location> <object>knife</object> </x> <status>suspended</status> </presence>	ArriveActivityAttribute sPresenceEvent
sendPresenceUsers Activity(Person, Activity,ActivityState)	<presence from='Maria-Agent@jabberserver'> <status>suspended</status> </presence>	ArriveActivityPresence Event

Table 1: SALSA API for notifying e-activity states

The sendUpdateActivity (Activity, Person) message makes it possible to register an e-activity or update a previous one. The e-activity is registered as a group item in the Broker roster. The sendPresenceActivityAttributes(Activity,ActivityAttributes) message enables e-activity agents to change the e-activity's attributes which are specified as an XML message that follows the SALSA message structure. Finally, the sendPresenceUsersActivity(Person, Activity, ActivityState) is used to indicate a change in the state of the user's activity. Thus, the roster contains the activated Activities, their states (such as executing, suspended, accomplished), and the Persons and the roles they play for carrying out the Activities. As illustrated in Scenario 2, an activity-aware system may present some of the e-activity attributes on the children's application to inform them regarding the state of the elder's activity. In this case, the activity "visiting the doctor" was registered in the roster with different persons (playing different roles) associated with it.

We extended the SALSA API to provide a set of communication messages that enables the system agents to access the CARe-ontology to register, update and retrieve e-activities through the Activity-aware SALSA service. The updateCARe(Activity,ActivityAttributes) is used to update an activity executed by a person, and its attributes such as the location in which the activity is being carried out. The requestCAReGraph(Activity||Action) is used to retrieve an activity carried out by the user and its execution context which is defined by its attributes.

6 Sample Application

6.1 Activity-aware Ambient Information Systems

In Scenario 1 and 2, we presented an activity-aware system designed to assist elders in different ways to manage their medication, i.e.: the system helps them to complete this activity, reminds them of some activity tasks or steps, or warns them when facing

risks associated with an activity. However, there are several challenges in designing the system user interfaces for the elderly, since we have to consider that they are characterized by presenting a natural cognitive decline, a decrease in their attention levels, and that they may consider that new technologies are complex to use. To address these challenges, we proposed Ambient Information Systems (AIS) since their user interfaces require minimal attention and cognitive effort to interpret the information presented, and are aesthetical pleasing[Mankoff,03]. AIS are embedded in the objects that elders currently use while carrying out their activities, which may enable elders to perceive that the activity-aware system is naturally integrated in their environment. Additionally, these objects may be more prone to attract their attention than other objects situated in their periphery. As presented in figure 1a) and 1b), the AIS designed for assisting the elderly are a Wearable Notification for Medication Adherence (WeNoMA) and an Ambient Medicine (AMe) Dispenser which present information according to the elders's location and activity status.

6.1.1 Wearable Notification for Medication Adherence (WeNoMA)

WeNoMA is a wearable device to provide elders with timely auditory and visual reminders for carrying out different routines related to their medication management:

- Reminders for taking their medicines which consists of an audio notification and information about the medicines to take: name of the medicines, doses, current time and an image associated with the health problem addressed by the medicine.
- Reminders for visiting the doctor, which present information regarding the appointment date and hour and an image of the medical card, that according to the case study results, they use for scheduling appointments.

6.1.2 Ambient Medicine Dispenser (AMe Dispenser)

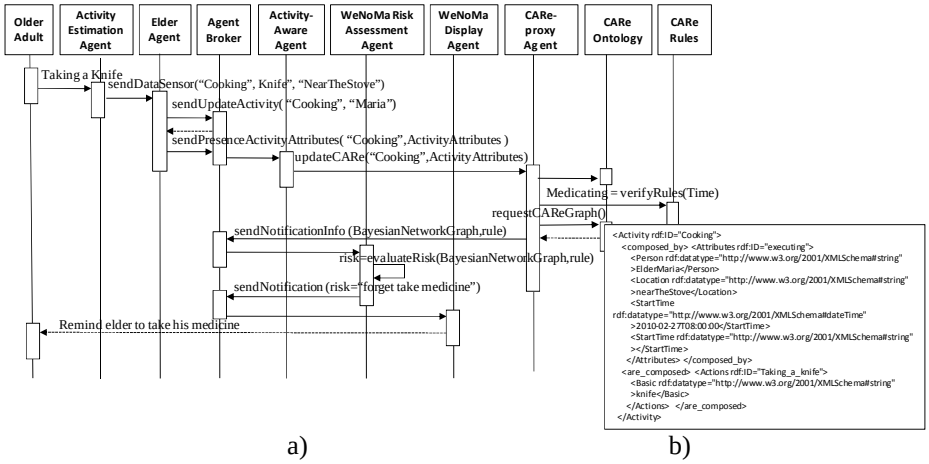
AMe is a medicine dispenser that assists elders while they are medicating. AMe provides elders with relevant information for medicating appropriately and to make them aware of the state of their daily medication routine. AMe consists of medicine containers which have geometric forms to indicate the number of times (frequency) that the medicine should be taken daily. For instance, figure 1b shows that Glibenclamida is in the triangular container since it has to be taken three times at day. Each container also informs about the medicine name, image associated with the health problem, and doses. AMe assists elders by informing which medicines need to be taken by using a blinking light to emphasize a container segment corresponding to the current frequency for taking the medicine doses; i.e., the first dose of Glibenclamida needs to be taken, once it is taken, the corresponding container segment stops blinking and remains on until the daily medication routine is completed. Thus, AMe provides awareness regarding the medicines that were taken during the day and how many doses are pending.

6.2 System Functionality

To illustrate how an Activity-aware System can be implemented with SALSA and their extended mechanisms for enabling an e-activity to hop among the devices that provide assistance to the elderly, we revisit scenario 1 which presents the functionality of the Activity-aware System to assist elders. The system architecture

(figure 3) presents the nodes attached to the SALSA middleware for implementing the Activity-aware System. These nodes contain SALSA agents that interact with the Activity-aware Service as showed in figure 6a and explained as follows:

While Maria is preparing her meals, her ActivityEstimationAgent notifies her personal agent (ElderAgent) the estimated activity and its attributes by using the sendDataSensor("Cooking","Knife","NearTheStove") SALSA message. Then, the ElderAgent registers the activity executed by Maria in the Agent Broker with the methods:sendUpdateActivity("Cooking","Maria") and sendPresenceActivityAttributes("Cooking",ActivityAttributes). This SALSA presence method notifies the Activity-Aware Agent the execution context of the activity which is determined by its attributes (location, used objects and current time of cooking). Then, the Activity-Aware Agent asks the CARe-proxy Agent to update the CARe ontology by using the updateCARe() SALSA method. The CARe-proxy Agent based on the activity execution context, verifies whether there is a condition which was met on the rule base (verifyRule()). According to the rules presented in figure 5, rule 1 and 2 created from the elder's medicine prescription and then rule 4 should be followed and therefore the elder should be reminded through the wearable device (WeNoMA). Thus, the CARe-proxy Agent requests the ontology graph (requestCAReGraph()) containing the current activity attributes as presented in figure 6b, and sends it to the RiskAssessmentAgent of WeNoMA which will evaluate whether the elder is prone to forgetting to take their medicines so that it can provide an audio reminder. To reach this end, the reasoning sub-component of the Risk Assessment Agent creates a Bayesian Network from Maria's CARe ontology graph to estimate the probability that she will forget to take her medicine given that she is in the kitchen doing other actions related to the cooking activity at the same time she needs to take her medicine. To apply the Bayes Rule we used Conditional Google Probabilities (CGP) [Perkowitz, 03] *to obtain the a-priori conditional probabilities. In this case, the CGP enabled the RiskAssessmentAgent to estimate the conditional a-priori probability that Maria takes her medicine given that she takes a knife, which is 2%. Thus, by using this CGP and by knowing from the medical literature that at least 26% of the elderly may forget to take their medicines, the Risk Assessment Agent applied the Bayes Theorem to determine that there is a 95% chance that Maria will forget to medicate. Finally, the Risk Assessment Agent informs this to the WeNoMADisplayAgent by sending a sendNotification() SALSA message containing the rules that govern how the wearable device should assist the elder. Thus, this agent provides an audio message reminder and presents visual information notifying the medicines to be taken. Advance inferring of this risk enables the activity-aware system to remind Maria to take her medicines without disturbing her by sending unnecessary reminders.*



a) b)

Figure 6: System sequence diagram. a) Agents interacting for reminding elder to medicate. b) Ontology retrieved for modulating the e-activity in the wearable device

7 Conclusions and future work

We have extended the SALSA middleware, and specially the CARe Knowledge Base, to help designers cope with the design issue of developing Ambient Intelligence environments that support the needs of elders when performing their Activities of Daily Living. For providing the context-awareness characteristic that these environments require for assisting their users, we follow the activity-based computing approach in which e-activity is the basic computational unit implemented with autonomous agents. Through the development of a couple of activity-aware systems we presented that when an activity is naturally and ubiquitously integrated into the environment where we live and into the objects we used on a daily basis it may migrate around these objects. Such migration requires new mechanisms that allow an e-activity, the computational unit of activity-aware systems, to move across devices and owners. SALSA autonomous agents enable the hoping and modulation of e-activities by accessing the Activity-aware Service and by managing the e-activity states and execution context through the SALSA API. Thus, we showed in this paper that SALSA autonomous agents can be used to address the complexities associated with implementing activity-aware applications, such as e-activity migration and modulation. Thus, the SALSA API and services help developers to concentrate on implementing the SALSA agents that capture the users' context and that provide the desired assistance to the elderly according to the rules that will govern their actions. We plan to evaluate the performance of the SALSA Activity-aware services and to assess the facilities provided by the middleware to enable the rapid development of activity-aware systems, and to predict its adoption by measuring the developers' perception of utility, ease of use and complexity.

Acknowledgements

We thank to the elders who participated in the case study and system evaluation; and to CONACyT for the scholarships granted to the students of this project.

References

[Bardram, 07] Bardram, J., Christensen, H. B.: Pervasive computing support for hospitals: an overview of the activity-based computing project. IEEE Pervasive computer 6(1), 44-51, 2007.

[Bochini,07] Bochini, C., Curino, C., Quintarelli, E.: A data-oriented survey of context models. In ACM SIGMOD Record, 36(4), 19-26, 2007.

[Bravo, 06] Bravo, J., Alaman, X., Riesgo, T., Ubiquitous Computing and Ambient Intelligence: New Challenges for Computing, Journal of Universal Computing Science, 12, 3, 233-235, 2006.

[Cabri, 2005] Cabri, G., Ferrari, L., Leonardi, L., Zambonelli, F.: The LAICA project: Supporting Ambient Intelligence via agents and ad-hoc middleware. IEEE Intern. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise, WETICE, 39-46, 2005.

[Cook, 06] Cook, D. J., Health Monitoring and Assistance to Support Aging in Place, Journal of Universal Computer Science, 12(1), 15-29, 2006.

[Corchado, 2008] Corchado, J.M., Bajo, J., Abraham, A.: GerAmi: Improving Healthcare Delivery in Geriatric Residences. IEEE Intelligent Systems, 23(2), 19-25, 2008.

[Gu,05] Gu, T., Keng-Punga, K., Da-Qing, Z.: A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications, 28, 1–18, 2005.

[Horgas, 04]. Horgas, A., Abowd, G., The Impact of The Technology on Living Environments for Older Adults. Technology for Adaptive Aging. The Nat. Academy Press, 230-252, 2004.

[Keng,09] Keng, P. H, Gu, T., Xue, W., Palmes, P. P., Zhu, J., Ng, W. L., Tang, C. W., Chung, N. H.: Context-aware middleware for pervasive elderly homecare. IEEE Journal on Selected Areas in Communications, Special issue on wireless and pervasive communications for healthcare, 27(4), 510-524, 2009.

[Kimel,07] Kimel J.: Exploring the nuances of Murphy's Law long-term deployments of pervasive technology into the Homes of Older Adults. ACM Interactions: 38-41, 2007.

[Lawton, 90] Lawton, M. P.: Aging and performance on home tasks. Human Factors, 32, 527-537, 1990.

[Mahmud,09] Mahmud, N., Vermeulen, J., Luyten, K., Coninx, K.: The five commandments of activity-aware ubiquitous computing applications. In Proceedings of Digital Human Modeling, LNCS 5620, Ed. Springer, 257-264, 2009.

[Mankoff,03] Mankoff, J., Dey, A. K., Hsieh, G., Kientz, J., Lederer, S., and Ames, M. (2003). Heuristic evaluation of ambient displays. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 169-176, 2003.

[Nardi,96] Nardi, B. A.: Context and consciousness: activity theory and human-computer interaction, MIT Press, Cambridge MA.

[Orwing,06] Orwing, D., Brandt, N., Gruber-Baldini, A.: Medication management assessment for older adults in the community, The Gerontologist, 661-668, 2006.

[Perkowitz, 04] Perkowitz, M., Philipose, M., Fishkin, K., Patterson, D.: Mining Models of Human Activities from the Web, Proceedings of The Thirteenth International World Wide Web Conference, ACM, 573-582, 2004.

[Philipose, 04] Philipose, M., Fishkin, K.P,. Perkowi, M.: Inferring Activities from Interactions with Objects. IEEE Pervasive Computing, 2004. 3(4), 50-57, 2004.

[Pousman, 06] Pousman, Z., Stasko, J.: A taxonomy of ambient information systems: four patterns of design. In Proceedings of the Working Conference on Advanced Visual interfaces (Venezia, Italy, May 23 - 26, 2006). AVI '06. ACM, 67-74, 2006.

[Rodríguez,05] Rodríguez, M. D., Favela, J., Preciado, A., Vizcaino, A.: Agent-based ambient intelligence for healthcare, AI Communications, 18(3), 201-216, 2005.

[Rodríguez,08] Rodríguez, M.D., Favela, J.: Evaluation of an Agent Framework for the Development of Ambient Computing. OTM Workshops, LNCS 5333, Springer, 374-383, 2008.

[Soldatos,07] Soldatos, J., Dimakis, N., Stamatis, K., Polymenakos, L..: A breadboard architecture for pervasive context-aware services in smart spaces: middleware components and prototype applications. Personal & Ubiquitous Computing, Ed. Springer, 11,193–212, 2007.

[Tentori,08] Tentori, M., Favela, J.: Activity-aware computing for healthcare. IEEE Pervasive Computing, 7(2), 51-57, 2008.

[Tentori, 10] Tentori, M., Rodríguez, M .D., Favela, J.: An Agent-based middleware for the design of activity-aware applications. To appear in IEEE Intelligent Systems.

[Zao, 08] Zao, J. K., Fan, S.C., Wen, M., H., Hsu, C. T., Hung, C.H., Hsu, S. H, Chuang, M.C., Activity-Oriented Design of Health Pal: A Smart Phone for Elder's Healthcare Support, EURASIP Journal on Wireless Comm. and Networking, Hindawi, 1-10, 2008.