A Personal Agent Supporting Ubiquitous Interaction

G. Cozzolongo, B. De Carolis, S. Pizzutilo
Dipartimento di Informatica
Università of Bari
{cozzolongo,decarolis,pizzutilo}@di.uniba.it

Abstract

This paper is concerned with the development of a MultiAgent System (MAS) supporting ubiquitous and personalized interaction with Active Environments. In particular, we propose a system in which a user may delegate her own agent to interact with the environment modeled as a set of agents providing services of different kind. In order to test the system, we implemented a smart to do list in which user can list what she intends to do. According to its content and to the current context situation, the user's personal agent decides to remind tasks or to proactively help the user in performing them.

Keywords
Personal agents, ubiquitous computing, smart environments.

1. INTRODUCTION

There are many different ways in which context information can be used to make applications more user friendly, flexible and adaptive especially in ubiquitous and pervasive computing where the context and usage needs change rapidly [1].

In Ubicomp computers fade into the background: the technology is present but invisible to users, realizing the vision of Weiser [2] in which the technology " is invisible, everywhere computing that does not live on a personal device of any sort, but is in the woodwork everywhere". Interaction between users and services offered by a smart environment is very complex as it can happen in every time and in different places. In this kind of situation adaptation to user and context features became very important.

Context-awareness then refers to the ability of a system of extracting, interpreting and using context information intelligently in order to adapt its functionality to the current context of use [3,4]. Considering the interaction between a user and a context-aware system, there are at least two aspects that are worth mentioning: **information presentation** and **service fruition**. As far as the first aspect is concerned, results of information services could be adapted not only to static user features, such as her background knowledge, preferences, sex, and so on, but also to more dynamic ones related to context (i.e. activity, location, affective state and so on) [5]. The second aspect regards execution of users tasks triggered by context features. For instance user's tasks present in a to-do-list or agenda could be proactively reminded or executed when the user enters in an environment or is in a situation in which those task are enabled [6,7]. Moreover, their execution can be contextualized according to available resources, location and so on.

This paper presents an approach to address this second research issue: taking advantage from user and context modeling for achieving effective ubiquitous interaction with services available in smart environments. A way to approach this problem is to take inspiration from the personal interface agents research area [8,9]. In this paradigm, the user may delegate a task to the agent that may operate directly on the application or may act in the background while the user is doing something else. An agent is, in this case, a representative of the user and acts on his/her behalf more or less autonomously. Moreover, it has to be able to communicate to the user in an appropriate way without being too much intrusive according to the context situation, user preferences, habits and needs. Then, importing this interaction metaphor in the Ubicomp vision, the ideal personal assistant, in addition, should exhibit a context-aware intelligence, doing exactly what the user expects him to do successfully in **the current context**.

Our work represents a first step in this direction. D-Me (Digital-Me) is a multiagent system composed at least two interacting entities: the **Environment** , a physical or logical place, in which various *services* are available, and one or more *mobile users,* interacting with ubiquitous services through a **Personal Agent**, his/her digital "alter ego". A relation between these two entities is represented by the task the user wants to perform in the environment and the services it can provide for accomplishing user's tasks. For this reason, in order to give users the possibility to delegate and control their D-Mes, when interacting with the environment, we developed, as a first prototype, a **Smart To-Do-List .**

A **To Do List** is a typical example of application that requires personalization and can take advantage from user and context modeling. Context-aware systems of this type remind the user of tasks based on the situational context. For example, if a user's to-do list contains the task 'buy food before going back home' and the user passes by a supermarket while going back home, then a useful context -aware reminder would notify the user to buy food. CyberMinder [7] and PDS[6] are examples of systems of this type. However, having an agent-based approach to the management of the To-Do-List allows to add the following feature: if there is a task in the agent that can be completely or partially executed by the agent the user may delegate its

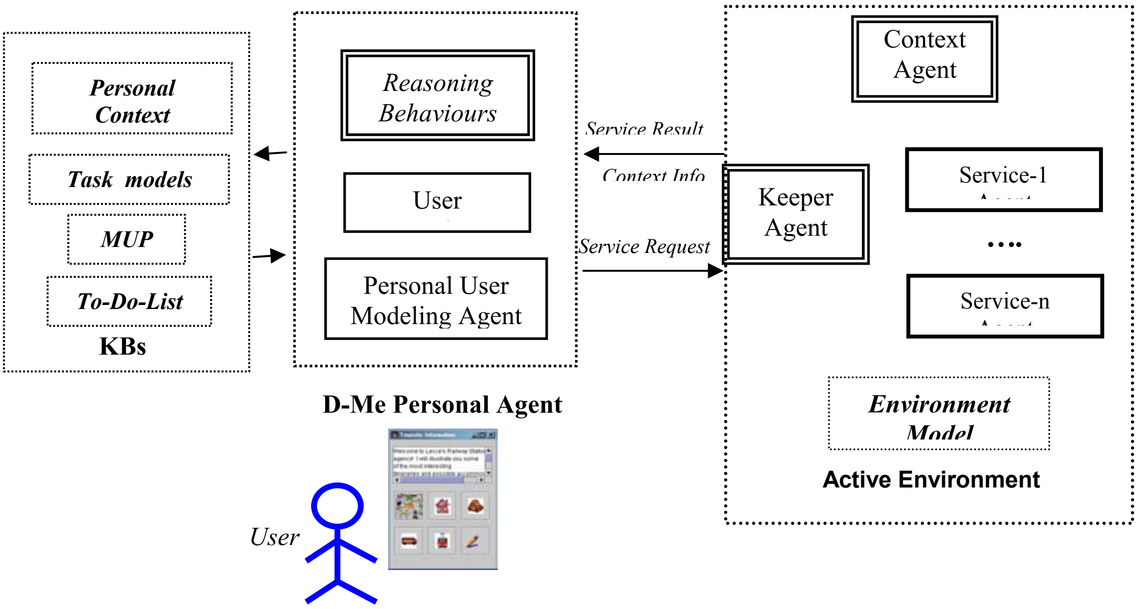


Figure 1: Layout of the D-Me Architecture

execution to the agent by giving it the appropriate autonomy level.

In this paper, we describe the D-Me architecture focusing on the description of the Personal Agent. The paper is organized as follows: Section 2 outlines some architectural requirements and describes the organization of the D-Me system. Section 3 focuses on the main features of the user Personal Agent outlining how it exploits several knowledge sources for supporting "personal" interaction with the Environment. In Section 4, conclusions and future work are discussed.

2. OUTLINE OF THE D-ME ARCHITECTURE

The D-Me MAS architecture is sketched in Figure1.

On one side, to support contextual service fruition, we developed a to-do-list application in which the user, through a friendly user interface, sets up a set of tasks to be performed in different context and environment and gives to his/her D-Me Personal Agent the autonomy to perform the task entirely or only in part [10]. When the user's agent is in presence of a smart environment, that can provide services useful to the execution of scheduled tasks, it requests, on user behalf, their execution by passing to the environment user related information, that can be used to get personalized results, according to privacy policies stated in the user profile. These results can be of various nature according to the nature of the performed service. For instance, a *confirmation* message is generated when a task step, on which the agent does not have the required autonomy, has to be executed; a *remind/notification* message is generated when some user tasks are executable in that context situation; user and context adapted *information presentations* is generated after, for instance, an information retrieval task; *assistance on task execution* is generated when help is needed in performing actions in the real or digital world, and so on.

To support this type of user and context aware service discovery, we modeled the smart environment as an organization of specialized agents:

- Service Agents, which are specialized in providing services and executing tasks;
- D-Me Personal Agents of other people present in the same environment;
- a Keeper Agent that acts as a Facilitator [11] and knows which (Personal and Service) agents are in the environment and, therefore, may provide on request environment-related information and the list of agents which could accomplish required services;
- a Context Agent that can handle data about the interaction context within that environment.

Users may interact with environment services in a remote way or by being physically in the environment and may move from one environment to another. Managing inter-environment communication is the task of the ***EnvKeeper*** that allows to handle a federation of environments.

From the implementation viewpoint, the D-Me has been developed according to FIPA specifications[12], by using JADE [13] as a toolkit, and though JAVA as programming language.

In particular, the Personal Agent, that we will see in more details later on, runs on a mobile device and has been implemented with JADE-Leap [14].

While the high-level communication protocols have been implemented using Agent Communication Language (ACL) messages, whose content refers to D-Me ontologies, the *service discovery* function has been developed using a framework for peer to peer communication, JXTA [15, 16]. Since FIPA has not yet defined a standard protocol for service discovery, this technology, joint to the JADE-Leap framework, allows the use of wireless communication protocols and supports the dynamic configuration of the system. The D-Me Keeper agent, based on the standard FIPA DF (Directory Facilitator) Agent, has been then modified to manage the service discovery protocols; this means that every time an agent registers in the environment, the Keeper will handle, besides standard information, also information about its public services. In this way, when an agent joins a platform, every other agent can be aware of its services.

In the rest of the paper, we will not going deep into this issue, because our aim is to show how the system works from the user point of view. Therefore, we will focus our description on the D-Me Personal Agent.

3. THE D-ME PERSONAL AGENT

D-Me Personal Agent is modeled as a BDI agent [17]; its reasoning mechanism is implemented as a *Thinkable Behavior* of the Agent Class. This type of cyclic behavior continuously checks if, given the current set of agent beliefs (mental state) and given its desires (goals), some intentions and plan can be triggered and executed.

At this stage of the implementation, we modeled the knowledge for achieving the following macro-desires:

– *execute totally or in part tasks specified in the user to-do-list*: this desire is quite complex and it is achieved by accessing the specification of the task in the user to-do-list and executing the correspondent task model according to the associated constrains (autonomy, user and context features).

– *create new tasks if required*: sometimes the context triggers the execution of tasks that were not explicitly stated in the to-do-list. In this case this desire become active given the appropriate level of agent's autonomy on that family of actions.

– *get user-related information relevant for adapting task execution*: in order to adapt task execution and to communicate results to the user appropriately, the agent needs to know information about the user. These information can be stored in a user profile or can be inferred.

– *get context-related information relevant for contextualizing task execution*: as for user related data, assessing the current context situation is important especially for triggering and adapting task execution.

– *communicate personalized results*: results of tasks can be of various nature (information presentation, reminders, notifications, and so on). The way in which the agent communicates to the user is adapted to user interests, knowledge, preference and so on, but also to context features.

Then, In order to achieve these desires, the Personal Agent exploits the following knowledge sources:

i) the ***to-do-list***, containing the description of the task and its constraints in terms of activation condition, priority, and autonomy level;

ii) the ***formal description of the task***, that the agent can use in order to execute it;

iii) the ***Mobile User Profile***, containing situational information about the user;

iv) the environment ***context situation*** (light, noise, etc.) from the Context Agent, and

v) the **personal context** situation reading the value of sensors that can be detected from devices that the user wear (heart beat monitors, temperature, etc.).

These joint sets of information forms the agent' set of beliefs and can be used to trigger opportune intentions formalized as "plan recipes". Planning is a fundamental and yet computationally hard problem [18], since D-Me is potentially running on different types of personal devices with limited computational power, predefined plan recipes seem to be a good compromise between flexibility and resource constraints.

To demonstrate our solution approach, we use the following scenario as a running example throughout the remainder of this paper:

The user enters into the to-do-list a very urgent task: "buy food before going home (18.00)". She finished working and is now driving home. D-me reminds her, using the car display as an output device, the task in the list that should be performed outside the office before coming back home. In this case, D-Me reminds her to buy food. The user acknowledges the message and drives to the supermarket, where she usually shops. When the user goes into the supermarket the agent shows the list of missing food and the related special offers on her PDA or telephone. The list is obtained by matching the items provided by the home fridge agent, that checks the fridge content using tagged objects technology, and the supermarket special offers (obtained using the service discovery technology of the supermarket keeper).

Let's see in more details how these knowledge sources are used by the agent to support context-aware interaction with the environment.

3.1 D-Me Autonomy

D-Me Personal Agent exhibits an autonomous behavior when achieving its desires that has to match, somehow, the user delegation type. In particular, in the context of ubiquitous computing, we recognized the need to model autonomy at different levels:

- **Execution Autonomy**: related to execution of actions (tasks, subtasks, request of services, and so on).

- **Communication Autonomy**: related to the level of intrusiveness in communicating to the user. Can the agent take the interaction initiative in every moment or there are constraints related to the user and the context? Then, it is necessary to determine how much a message can be intrusive in a certain context.
- **Personal Data Diffusion Autonomy**: it is related to the autonomy of performing tasks requesting the diffusion of personal data like those contained in the user profile.
- **Resource Autonomy**: the agent may use critical resources of the user in order to executed delegated tasks (i.e. credit card number, time to schedule appointments).

Each dimension has an associated value that vary from "null" to "high" in a 5 values scale. The "null" value represents the absence of autonomy, the system has to execute what explicitly requested by the user. It cannot infer any information or decide to modify task execution without explicitly asking it to the user. The opposite value, "high", represents the maximum level of autonomy that gives to the agent the freedom to decide what to do always according to constraints imposed by the user (i.e. budget limits). The other values determines an incremental growing of the autonomy in making decisions and inferring information [10].

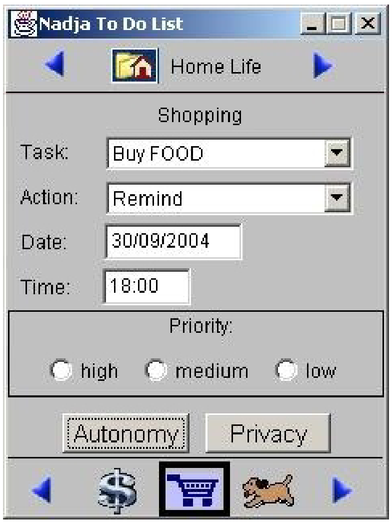
Initially, as we will see later on, the user sets explicitly the autonomy level for a task in the to-do-list. During the interaction, autonomy levels are revised according to the type of feedback the user provides to the agent: ***positive*** feedback enforces the autonomy on that category of task , ***negative*** one reduces it. We are aware this is a simple mechanism, however it will give us the possibility to conduct a further study aiming at learning which is the most appropriate relation between the agent's level of autonomy and the type of user delegation on a category of tasks.

3.2 To-Do-List and Task Models

In order to give to the user personal agent the capability to reason on this information, it is necessary to specify the entry in the To-Do-List in terms of **type** or family of **task** , **environment** and **context information** relevant for task execution, **user** related **preferences**, agent's **autonomy**. On this purpose, we developed an interface in Java running on a PDA that enables user to input this information in a quite simple way (Figure 2). A To-Do-List entry is then translated and formalized in XML and stored in the set D-Me Knowledge Bases. An example of entry corresponding to "buy food before coming back home" is the following:

```
<Knowledge
xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-
instance"
xsi:noNamespaceSchemaLocation="C:\DmeSystem\dati\
Knowledge.xsd" slotName="ToDoList">

<Task slotName="taskDefinition" id="1" name="buy"
key="food action="remind" " date=2509041800"
belongingScope="homelife" environment="all"  p-
env="supermarket" priority="high" what="food
list" when="before" whenevent="going back home"
remindBefore="1755" nextOk="3" nextError="4">

<Autonomy slotName="autonomy" execution="high"
communication="high" personalData="middle"
resourcesExploitation="low" />

</Task>

….

</Knowledge>
```

This specification states which is the task name, the type of associated D-Me action to be performed when the contextual situation triggers it (*remind* in this example), the scope of the task (*homelife*) that can be used to trigger user preferences in that scope, the *environment* in which the task should be reminded and the one in which the task should be performed (*p-env*). Then there are other information regarding the priority, the deadline and the type of agent *autonomy* on that task.


Nadja To Do List

Home Life

Shopping

Task: Buy FOOD

Action: Remind

Date: 30/09/2004

Time: 18:00

Priority:

○ high ○ medium ○ low

Figure 2. A snapshot of the To Do List Interface.

In this example, the agent has an high execution and communication autonomy, a medium autonomy in communicating personal data to the environment and low autonomy on resource exploitation (in this example this is translated in the fact that the agent cannot buy and pay autonomously the food unless it is explicitly authorized by the user).

When user and context features triggers one of the tasks present in the user To-Do-List, the agent's desire of executing a task is achieved by selecting the appropriate plan in the D-Me KB.

In this case the *Remind(U, Do(Task , env, p-env Cti))* plan is selected. In this case, *U* denotes relevant user features, *Task* denotes *Buy(food)*, *env* the environment in which the remind can be notified (all), *p-env* the environment in which the user task can be performed (supermarket) and *Cti* represents the context at time *ti.*

3.3 P.U.M.A.: Personal User Modeling Agent

Mobile personalization can be defined as the process of modeling contextual user-information which is then used to deliver appropriate content and services tailored to the user's needs. As far as user modelling is concerned, a mobile approach, in which the user "brings" always with her/himself the user model on an personal device, seems to be very promising in this interaction scenario [19]. It presents several advantages: the information about the user are always available, updated, and can be accessed in a wireless and quite transparent way, avoiding problems related to consistency of the model, since there is always one single profile per user.

Based on this idea, in the context of our research on personalization of interaction in ubiquitous computing [20, 21], we have designed and implemented a Personal User Modeling Agent (PUMA).

In developing its architecture we considered the following issue: a personal device is used mainly in situations of user mobility. Normally, when the user is in more "stable" environments (i.e. home, office, etc.) he/she will use other devices belonging to that environment (i.e. PC, house appliances, etc.). In this view, the personal device can be seen as a "satellite" of other "nucleus" devices that the user uses habitually in his/her daily life. Then, the PUMA has to be able to handle this nucleus-satellite relation.

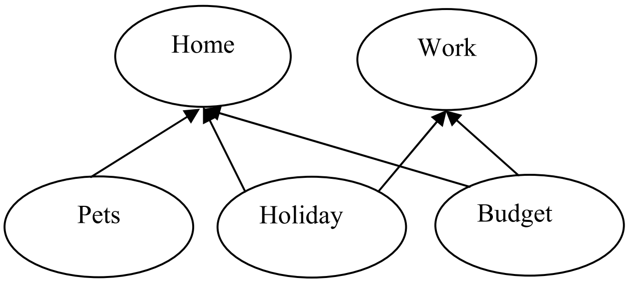


Figure 3: An example of hierarchical User Model

With this aim, instead of implementing a truly mobile agent, the PUMA is cloned and lives on all the user platforms/devices. However, although the chosen approach simplifies the implementation, it requires transferring knowledge needed for user modeling and opens consistency problems in maintaining a global image of user preferences, interests, habitual behavior, etc. In our approach, user models are organized in a hierarchy [22] whose nodes represent relevant interaction environments, task families, interest groups (Figure 3).

In particular, the roots of the hierarchy represents user modeling scopes (interaction environments). Each node in the hierarchy represents a subset of user model data relevant to the specified domain, task, etc. Then the PUMA accesses and reasons on the Mobile User Profile portion that is in focus according to the user task and environment.

This approach presents the main advantages of decreasing the complexity of representing an unified view of the user profile even if it requires particular attention in structure modelling and decomposition.

In another project we are testing how the same approach could be implemented using a hierarchy of Bayesian network instead of MUPs allowing in this way a better treatment of uncertainty that is typical of ubiquitous computing [23].

As far as ***representation*** is concerned**,** beside considering static long term user features (age, sex, job, general interests and so on) it is necessary to store information about more dynamic "user in context" features.

For instance, the fact that a user, when is shopping at the supermarket, *buy cookies only when there is a 3x2 special offer* is related to a contextual situation. If we want to give to the user PUMA the capability to reason on this type of facts, we need a representation language rich enough to fomalize user properties related to contextual situation, understandable potentially by every environment, flexible and compact enough to be stored on the user personal device. In a first version of D-Me we developed our own ontology for describing mobile user profiles, however, since it was not the main aim of our research, in this second version of the prototype, we decided to adopt UbisWorld" [24, 25] language as user model ontology of our Personal Agent. In this way we have a unified language able to integrate user features and data with situational statements and privacy settings that better suited our need of supporting situated interaction. This language is rich enough to deal with the representation and provide privacy-protection features. It allows representing all *concepts* related to the user by mean of the UserOL ontology, to annotate these concepts with situational statements that may be transferred to an environment only if the owner user allows this according to privacy settings. An example of a situational statement is the following:

```
</SituationalStatement>
<SituationalStatement version="Full_0.1">
< content
<subject><UbisWorld:Nadja /></subject>
<predicate><UserOL:buying cookies /></predicate>
<predicate-range><UserOL:normal,specialoffer,
3x2/> </predicate-range>
<object>special offer <object>
</content>
</content>
<restriction>
< location>supermarket<location>
</restriction>
<meta>
<owner><UbisWorld:Nadja /></owner>
<privacy><UbisWorld:friends /></privacy>
<purpose><UbisWorld:commercial /></purpose>
<retention><UbisWorld:short /></retention>
<viewer><UbisWorld:X-Supermarket /></owner>
<evidence>not-specified</evidence>
<confidence>high</confidence>
</meta>
</SituationalStatement>
```

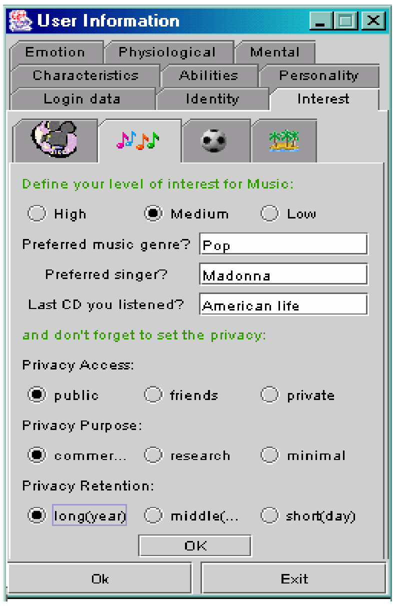
This approach can be used to represent some parts of the real world like an office, a shop, an house or an airport. It represents persons, objects, locations as well as times, events and their properties and features.

Information contained in the MUP together with the context situation and the task list, deriving by accomplishing the relative Agent desires, may be used by D-Me to decide how to adapt its behavior and which data to pass to the environment.

User preferences, interests, etc. are collected by the PUMA in two ways:

- using a graphical interface (**Figure 4**) in which the user can explicitly insert her preferences and related privacy settings regarding particular domains,
- other information (i.e. temporary interests) can be derived when the user insert a task in the To-Do-List.

User feedback and actions in the digital and real world may reproduce changes in the user model. The PUMA observes the user actions: when new information about the user can be inferred, it updates or adds a new slot in the MUP and sets the "confidence" attribute of that slot with an appropriate value that is calculated by the weighted average of all the user actions having an impact on that slot. The confidence attribute may be set to low, medium and high.



Define your level of interest for Music:

○ High ● Medium ○ Low

Preferred music genre? Pop

Preferred singer? Madonna

Last CD you listened? American life

and don't forget to set the privacy:

Privacy Access:

● public ○ friends ○ private

Privacy Purpose:

● commer... ○ research ○ minimal

Privacy Retention:

● long(year) ○ middle(... ○ short(day)

Figure 5. An interface for initial setting of the PUMA.

Even if we have chosen the mobile approach, we cannot assume that the user will have with her/himself an handheld device and this type of device still presents hardware-related limits (capacity, computational speed, battery,…).

At this aim, in D-Me the PUMA could be stored on a Remote Server trusted by the user [26]. In the near future these technological constraints will be overcome by the spread of many personal an powerful device [27,28]

3.4 Context Information

Both entities, D-Me Agents and the Environment, need to sense and elaborate context information. In our approach, **Context** is grounded on the concept of "user task executable in an environment". Therefore, given a task in the user to-do-list, once the user has been classified according to the strategy of the UM component, its execution and results can be influenced by the context in which the interaction occurs and, in particular, by:

– ***static environment features***: scope (daylife, social relations, budget, etc..), physical features, such as desciption of objects relevant for interaction, type of environment (public, private).

– ***dynamic environment features***: for instance noise light level and tagged object;

– ***dynamic user features***, that identify the physical and social surroundings of the user that can be derived by specific data sensors (emotional state, location, activity the user is performing, time, ...);

– ***device*** employed.

At the present stage of the prototype, we do not work on hardware sensors. They will be realized in the next stage. At the moment we simulate their values through an interface that sends them to dedicated **Sensors Agents**, which communicate relevant changes to the **Context Agent** that knows the global context situation at the considered time. The context situation relevant at time *ti* is represented in an XML structure compliant to the D-Me context ontology.

3.5 Interacting with the user

The Communication Behavior of the Personal Agent is used to interact with the user for communicating results of tasks or for asking information/confirmations required for task execution. We consider the following families of communication tasks:

- ***request for input***. If, for instance, the to-do-list includes a task that requires additional information to be executed.

- ***information provision***: Information may be presented when explicitly requested by the user or proactively prompted by D-Me because related to the current user task. In our scenario the supermarket special offers will be displayed as a consequence of the service discovery task.

- ***request for confirmation:*** if a task involves a step that requires a D-Me action and the agent does not have a full autonomy on that task, then the agent will ask the user for confirmation before performing it.

- ***notification messages***. Proactive task execution is notified by D-Me, for instance, in the previous case, if the agent has the autonomy to perform an action then it will not ask for permission and will just notify it.

- ***remind messages***. This is the typical message generated for the shopping task in our example.

User and context related factors are taken into account in generating the communication about a task in the following way [29]:

1. **user preferences and features**: results of information provision tasks are filtered, ordered and presented according to what has been inferred about the user starting from her profile data (interest, know-about, know-how). Possible user disabilities are taken into account for media selection.

2. **activity**: this influences information presentation as follows. If the user is doing something with a higher priority respect to the one of the communication task, then the message is postponed until the current activity ends. If the communication regards the current activity, media used in the message take into account the available body parts. Therefore, a voice input is preferable to a textual when, for instance, the user is running with her/his PDA asking for information about the next train to catch.

3. **location** of the user in the environment: texts, images and other media may be selected according to the type of environment (public vs. private, noisy vs. silent, dark vs. lightened, etc.) in which the users are and, more precisely, to their relative position to relevant objects in the environment.

4. **emotional state**: factors concerning the emotional state influence the level of detail in information presentation (short messages are preferred in stressing situation), the intrusiveness (bips and low priority messages are avoided when the user is already nervous), and the message content. For instance: if a user requests information in conditions of emergency, the agent will have to avoid engendering panic, by using reassuring expressions or voice timbre [30].

5. **device**: the display capacity affects the way information is selected, structured and rendered. For instance, natural language texts may be more or less verbose, complex figures may be avoided or substituted with ad hoc parts or with written/spoken comments.

To accomplish the communication task, the agent applies the following strategy: starting from XML-annotated results of a Service Agent, decides how to render them at the surface level taking into account the rules described above encoded in XSL.

4. DISCUSSION AND FUTURE WORK

Effective ubiquitous interaction requires, besides techniques for recognising 'user in context' features, a continuous modeling of both the user and the context. Therefore, ubiquitous computing systems should be designed so as to work in different situations that depend on several factors: presence of a network connection, characteristics of interaction devices, user location, activity, emotional state and so on. However, in the near future, the network connectivity will be no more a problem, and we will not be worried about this constraint, as we are going towards an "interconnected world". Moreover the spread of technologies, such as for example RFID, will render the information about the context very rich and easy to use [31].

This work represents a step towards supporting personalized interaction between mobile users and a smart environment. Every user is represented by a D-Me Agent that, according to the content of her/his "To Do List", performs tasks on the user behalf by negotiating services with the smart environment.

Since the interaction happens through a personal agent, we started to consider the "delegation-autonomy" adjustment necessary for achieving cooperation between the user and his/her representative. However, more work in understanding how the user feedback influences the level of autonomy especially when this feedback is implicit (until now we considered only explicit feedback).

Moreover, as RFID are taking a key role in ubicomp we are investigating how to use them in such a system, so as to "sense" the active tagged object. Those kind of object are part of the context and can influence the execution of several tasks as well as other information.

ACKNOWLEDGMENTS

We thanks students who cooperated in implementing the prototype described in this paper: in particular, Ignazio Palmisano, Luigi Iannone, and Roberto Tagliento. Finally, we thank Fiorella de Rosis to which we owe several fruitful ideas underlining this work.

REFERENCES

1. Greenberg, S. (2001). Context as a dynamic construct. Human-Computer Interaction, 16.
2. M. Weiser. Some computer science issues in ubiquitous computing. Commun. ACM, 36(7):75-84, 1993.
3. Chen G. and Kotz. D. A Survey of Context-Aware Mobile Computing Research. Technical Report TR http://citeseer.nj.nec.com/chen00survey.html.
4. Dey A. K. Understanding and Using Context. Personal and Ubiquitous Computing 5 (2001) 1, 4-7.
5. Ardissono L, A. Goy, G. Petrone, M. Segnan and P. Torasso. Ubiquitous user assistance in a tourist information server. Lecture Notes in Computer Science n. 2347, 2nd Int. Conference on Adaptive Hypermedia and Adaptive Web Based Systems (AH2002), Malaga, pp. 14-23, Springer Verlag 2002.
6. Byun H.E. & Cheverst K. User Models and Context-Awareness to Support Personal Daily Activities. Workshop on User Modelling for Context-Aware Applications. 2002.
7. Dey A.K. and Abowd G.D. CyberMinder: A Context - Aware System for Supporting Reminders. Proc. Symposium on Handheld and Ubiquitous Computing, Bristol. (2000).

8. Maes, P. "Agents that Reduce Work and Information Overload," Communications of the ACM, Vol. 37#7, ACM Press, 1994.

9. Lieberman H. and Selker T. Out of Context: Computer Systems That Adapt To, and Learn From, Context. IBM Systems Journal, Vol 39, Nos 3&4, pp. 617-631, 2000.

10. Falcone R., Castelfranchi: Tuning the Collaboration Level with Autonomous Agents: A Principled Theory. AI*IA 2001: 212-224.

11. http://www.ai.sri.com/~cheyer/papers/aai/oaa.html

12. http://www.fipa.org.

13. http://sharon.cselt.it/projects/jade/

14. http://leap.crm-paris.com/

15. http://www.jxta.org

16. Michael Pirker, Michael Berger, Michael Watzke,, An Approach for FIPA Agent Service Discovery in Mobile Ad Hoc Environments , Workshop on Agents for Ubiquitous Computing., Ubiagent04.

17. A. S. Rao and M. P. Georgeff, BDI-agents: from theory to practice, in "Proceedings of the First Intl. Conference on Multiagent Systems", San Francisco, 1995.

18. Tom Bylander. The computational complexity of propositional STRIPS planning. Artificial Intelligence, 69:161–204, 1994.

19. Kobsa A., Generic User Modeling Systems. UMUAI vol. II nos.1-2 pp.49-63. Kluwer Academic Publisher. 2001.

20. Cavalluzzi A., De Carolis B., Pizzutilo S., Cozzolongo G.: Interacting with embodied agents in public environments. AVI 2004: 240-243.

21. Cozzolongo, G., De Carolis, B., Pizzutilo, S.. Supporting Personalized Interaction in Public Spaces. In Proceedings of the Artificial Intelligence in Mobile Systems 2004. (Ubicomp Workshop). Baus J., Kray, C., Porzel, R. (Eds.). Nottingham, UK, 2004.

22. Samulowitz M. Designing a Hierarchy of User Models for Context-Aware Applications. Position Paper at www.teco.edu/chi2000ws/papers/23_samulowitz.pdf

23. Jameson A. Modeling Both the Context and the User. Personal and Ubiquitous Computing. Vol 5. Nr 1. Pp 29-33. 2001.

24. Dominik Heckmann: Ubiquitous User Modeling for Situated Interaction. User Modeling 2001: 280-282

25. Heckmann, Ubis World, www.u2m.org

26. De Carolis B., Pizzutilo S., Palmisano I. and Cavalluzzi A. A Personal Agent Supporting Ubiquitous Computing. UM'03 9th International Conference on User Modeling. June 22-26, 2003.

27. www.vodafone.com

28. Roy Want, http://www.intel.com/research/exploratory/personal_server.htm

29. De Carolis B., de Rosis F. and Pizzutilo S. Adapting Information Presentation to the "User in Context". IJCAI 2001 Workshop on AI in Mobile Systems, Seattle, 2001.

30. A Cavalluzzi, B De Carolis, V Carofiglio and G Grassano:
Emotional dialogs with an embodied agent.
In P Brusilovsky, A Corbett and F de Rosis (Eds): "User modeling '03".

31. Pederson, T. (2003). From Conceptual Links to Causal Relations -Physical-Virtual Artefacts in Mixed-Reality Space. PhD thesis, Dept. of Computing Science, Umeå university, report UMINF-03.14, ISSN 0348-0542, ISBN 91-7305-556-5. http://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-137.