An Augmented Campus Design for Context-aware Service Provision

Alessandro Genco
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 7028517
genco@unipa.it

Salvatore Sorce
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 7028526
sorce@unipa.it

Giuseppe Reina
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
reina@studing.unipa.it

Giuseppe Santoro
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
santoro@studing.unipa.it

Rosario Messineo
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
messineo@studing.unipa.it

Paolo Raccuglia
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
raccuglia@studing.unipa.it

Laura Lovecchio
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
lovecchio@studing.unipa.it

Giorgia Di Stefano
Dipartimento di Ingegneria Informatica
Viale delle Scienze, edificio 6
90128 Palermo - ITALY
+39 091 6598043
distefano@studing.unipa.it

ABSTRACT

This paper deals with the design of a multi-modal system for pervasive context-aware service provision and human-environment interaction in augmented environments by the use of Personal Digital Assistants (PDA) or SmartPhones. The system enables mobile devices and remote displays to perform as interaction devices with pervasive applications which run on a dynamically composed server network. Visual interaction for service setup and provision are driven by appropriate graphical interfaces and XML-based protocols, which are dynamically composed according to the type of service and to the user current position by means of a mobile agent-based framework. The paper discusses both protocols, hardware and software system components. The first part of the document gives a general description of the system, which is managed by an entity-driven organization in augmented reality.

The mobile and reference devices of the system framework are then discussed, along with the mobile agent software which is used to manage connections among them and with system entities.

The paper also gives some details about the ad-hoc protocols for entity interaction. Next, a case study is discussed dealing with service provision in a campus augmented environment which has been arranged according to service requirements. Finally the paper discusses some user experiences while using trial services.

SIGUCCS'05, November 6–9, 2005, Monterey, California, USA.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Artificial, augmented, and virtual realities.

General Terms

Management, Experimentation, Human Factors.

Keywords

Pervasive systems, smart environments, context-aware services, mobile devices.

1. INTRODUCTION

Smart phones and PDAs are likely to be enabling personal mobile devices to implement the ubiquitous computing vision. Mainly due to their small size, such devices are still limited as for human-computer interaction. Embedded displays and speakers are far from being suitable for multimedia slideshows indeed, even for quite small audiences. On the other hand, besides data and voice communication or sensing capabilities, personal mobile smart devices are often equipped with general-purpose processing systems, thus being perfect for interaction with augmented reality. Such devices can be made suitable to operate as remote control interfaces of some application running somewhere in the environment. This way the environment is mainly devoted to store information and services people are to be provided with. To this end, personal mobile devices can be exploited both for requesting information and services, and for carrying user profiles which can be used to select specific information or services.

The system we propose is designed to supply people with ad-hoc information and high-level services by means of personal mobile devices exploited as adaptive human-environment interfaces. The access to the system services requires a very small overhead for users, in terms of required technical skills and software to be installed and used. Services and access modes definition are also very straightforward, and modifications can be made on-line. Last but not least, the hardware needed to access system services is something people commonly use in their everyday life.

2. RELATED WORKS

There are several existing projects dealing with the use of personal mobile devices as interfaces for some hidden application logic [11]. Authors in [10] implemented a WWW browsing system for cellular phones which allow users to browse HTML pages on a remote display. The central server creates and sends an operational page to the mobile terminal. Users can scroll the page on the display and follow selected hyper-links.

As far as service provision is concerned, the Iceberg project [12] deals with communication technologies integration. Cellular phones as well as PCs and network devices are used to seamlessly integrate data and voice.

Museums, meeting rooms, and classrooms are typical environments where personal mobile devices can be used as I/O interfaces to access pervasive services. There are several projects whose goal is to support users in such pervasive environments by providing them with ad-hoc information or media on mobile devices. In this field, Busetta et al. [3] proposed a system where agents are used for dynamic information generation within active museums, while Chen et al. [4] implemented an agent-based system for teaching support in a smart classroom. All these projects require the use of ad-hoc software for mobile devices.

Dealing with context-aware service provision in augmented reality, it must be taken into account the relevant role of the position. We implemented a low-cost Bluetooth based positioning system to this end, which relies on Link Quality measurement between mobile and reference devices [1]. Good results came from our experiments which were aimed at improving accuracy and optimizing the fixed devices layout by means of genetic and simulated annealing algorithms [8].

3. SYSTEM ARCHITECTURE

The framework of our proposed system relies on the HAREM model which describes all physical and logical entities involved in augmented environments as *hybrid entities* [7]. Their task is to supply people with selected information or services by means of personal mobile devices, such as PDAs or SmartPhones.

HAREM hybrid entities rely on a three layer stack, each hosting a different entity projection: a semantic projection for semantic interaction, knowledge maintenance and knowledge management; a middleware projection which has in charge entity implementation according to some development platforms; a physical projection for physical interaction and physical resource management. We implemented HAREM entities as intelligent agents [6], with some application specific roles in addition to some basic roles which were thought for any kind of entity in augmented reality.

HAREM-compliant hybrid entities interact with each other by means of peer layer "horizontal" communication, which take place through services supplied "vertically" by lower level layers.

3.1 The Environment Structure

The needed support hardware framework is made of fixed devices networked (wired or wireless), hidden in the environment. The architecture of such framework is hierarchical, and the environment is divided into logical areas, each for specific task accomplishment within the network, and each belonging to a given hierarchical level. Two or more logical areas can be connected to obtain a simple subnet that can be connected to other subnets to obtain a larger network. Each area or subnet can be connected to, or disconnected from, the network at any time, thus leading to a dynamic level distribution. Each logical area acts as an autonomous entity when it is disconnected from any other subnet, and users within the area can only access local services.

There are two area types: Service Areas (or 1^{st} Level areas), and Resource Areas (or N^{th} Level areas, N>1). The Service Area (SA) is the smallest entity within the framework. It is composed of a number of networked Access Points (APs) (See Figure 1: A, B, and C), and its physical size corresponds to the coverage of its AP. Users can access information stored in the SA central server (See Figure 1, D) through APs used as anchor nodes for mobile devices (See Figure 1, F).

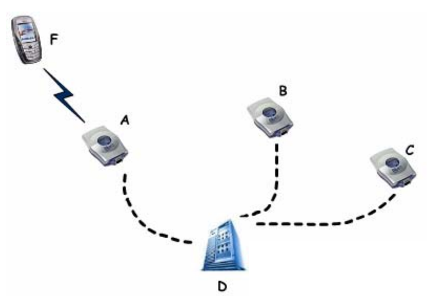

Figure 1. Service Area template.

A Resource Area (RA) is composed of a federation of one or more lower level areas, whose coordination is managed by a Control Area Server. The RA topology depends on its hierarchical level. A 2^{nd} level RA contains one or more SAs, while higher level RAs contain lower level RAs and SAs. The highest level RA is called "Root Area"; it contains all the lower level areas and does not have any higher level RA. Figure 2 shows an example environment.

The software framework is based on hybrid entities implemented as mobile agents by means of the JADE development tool [2]. All servers run a mobile agent platform, a database to store information, and a web server to access information. Each platform performs specific tasks by means of appropriate agents depending on the kind of area in which it runs. Common elements of the system that can be found in a platform, depending on the area type, are:

- one *LoggerEntity* to manage users login and for routing service;
- one *SetupEntity* for the start-up phase and to setup connections in order to register the area within the network.
- one *UserEntity* for each user logged in within the area;
- one *ConnectionEntity* for each AP to manage connections between mobile devices and servers;
- one or more *ServiceEntities* for each kind of service the Area provides.

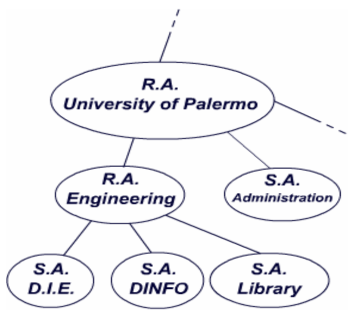

Figure 2. A Resource Area template for a university environment, with hierarchical areas representation.

System administrators do not have to setup an entire network at the same time. They can start the system with some subnets and then link them later. In particular, a SA can be linked to a RA in order to extend the network. Once the linking phase is successfully ended, users within a SA can access services supplied by another SA. In the following we describe the linking protocols, which are implemented by means of communicative acts among involved entities.

4. IMPLEMENTATION DETAILS

SetupEntities manage inter-platform communication during the linking phase. SetupEntities store the area type, the services list, the IDs of lower level SetupEntities, and the ID of higher level SetupEntity. Such information is stored in XML files, which are used by SetupEntities of other areas to discover local available services. XML files are used to store the way users access local services too. Communication among entities takes place by means of FIPA-ACL messages [5], composed according to an appropriate communication protocol.

Regardless of the area type, each area platform stores information needed to access the available services in a local XML file, whose name is `area.xml`. In this case, the file stores the list of locally available services and information on how these services can be accessed.

The `area.xml` file of a RA is created by the local SetupEntity during the linking phase with another area. In this case the file stores the links to `area.xml` files of associated areas. In other words, both SAs and RAs store an `area.xml` file. This file is used in different ways: in a SA it is used to store information needed to access local services, while in a RA it is used to list services provided by lower level SAs.

A SA server stores another XML file, whose name is `index.xml`. This file is created and managed by SetupEntities, and it is used by the client-side application to show available options for that Area. When a SA is started, `area.xml` and `index.xml` files store the same contents. The `index.xml` file will be changed according to the current SA linking status.

The link between a SA and a RA takes place by means of FIPA-ACL message exchange between SetupEntities of linking areas. In the following we briefly describe the message exchange protocol and the XML files format. To this end we will refer to a specific environment in which we carried out some experiment.

4.1 Service Area Linking Protocol

We implemented a trial version of our system at our University. There we carried out some experiments to test our design choices. The environment is made of departments, each with specific feature and services (research, teaching, secretarial), and each with some kind of user profiles (professors, students, employees). Users could be supplied with these services:

- documents request;
- exam subscription;
- access to teaching information, such as course time, exam dates, etc.;
- teaching requests, to be directly addressed to the mobile device or to be sent to an e-mail address.

Such services can be accessed by means of personal mobile devices, which are communication-enabled and are equipped with some application execution environment.

The augmented environment provides people moving around with information and services by means of an appropriate set of devices which can be split into two groups:

- mobile devices, like PDAs, SmartPhones, notebooks, or even laptop computers, provided that they are communication-enabled and equipped with some application execution environment.
- reference devices, like PCs exploited as database or servers, and wireless access points to connect mobile devices with fixed part of the system.

Suppose an initial network with two SAs, for the Library and the D.I.E. (Department of Electrical Engineering) departments, connect to a RA representing the "Engineering" faculty (See Figure 3).

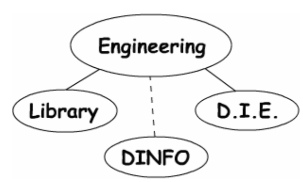

Figure 3. Trial environment initial configuration

The steps leading to the linking of the DINFO (Department of Computer Engineering) SA to the network are as follows. The SetupEntity of DINFO SA contacts the SetupEntity of Engineering RA with the following ACL message:

```
Sender: SetupEntity@DINFO
Receiver: SetupEntity@Engineering
Performative: INFORM
Protocol: ADD_PLATFORM
Content: DINFO@http://dinfo.unipa.it/xml/Area.xml
```

When the SetupEntity@Engineering receives the message, it stores the sender's ID in the list of lower level areas. The message content is used to create an XML node:

```
<item title = "DINFO">
  <link url = "http://dinfo.unipa.it/xml/Area.xml"/>
</item>
```

Such a node is included in the local `area.xml` file, thus obtaining:

```
<ui>
  <list title = "Engineering">
  <item title = "D.I.E.">
    <link url = "http://die.unipa.it/xml/Area.xml"/>
  </item>
  <item title = "Library">
    <link url = "http://biblioteca.unipa.it/xml/Area.xml"/>
  </item>
  <item title = "DINFO">
    <link url = "http://dinfo.unipa.it/xml/Area.xml"/>
  </item>
  </list>
</ui>
```

The SetupEntity@Engineering at last sends the following message to the SetupEntity@DINFO:

```
Sender: SetupEntity@Engineering
Receiver: SetupEntity@DINFO
Performative: INFORM
Protocol: ACK_PLATFORM
Content: http://ingegneria.unipa.it/xml/Area.xml
```

The last communicative act is an ACL message sent by SetupEntity@Engineering to other lower level SetupEntities:

```
Sender: SetupEntity@Engineering
Receiver: SetupEntity@DIE; SetupEntity@Library
Performative: INFORM
Protocol: UPD_CONTENT
Content: http://ingegneria.unipa.it/xml/Area.xml
```

This way, all lower level SAs download the `area.xml` file, whose url is shown in the "Content" section of the last message, and overwrite the local index`.xml` file with the new data.

Suppose a RA has to be connected to another RA. Figure 4 shows the case where the Engineering RA has to be connected to the "University of Palermo" RA. The communicative acts layout between the SetupEntity@UniversityOfPalermo and the SetupEntity@Engineering is the same as was previously described. At the end of the linking phase, the SetupEntity@Engineering updates its `area.xml` file by adding a "More services…" node, and then it sends an UPD_CONTENT message to the lower level SAs. The updated `area.xml` file of the RA Engineering will be as follows:

```
<ui>
  <list title = "Engineering">
    +<item title = "D.I.E.">
    +<item title = "Library">
    +<item title = "DINFO">
      <item title = "More services...">
    <link url = "http://www.unipa.it/xml/Area.xml"/>
  </item>
  </list>
</ui>
```

Areas shown in Figure 4 operate in different ways. In particular, Literature RA will do any change in its files. Administration SA will update its `index.xml` file with the content of the `area.xml` file whose url is shown in the "Content" section of the received ACL message.

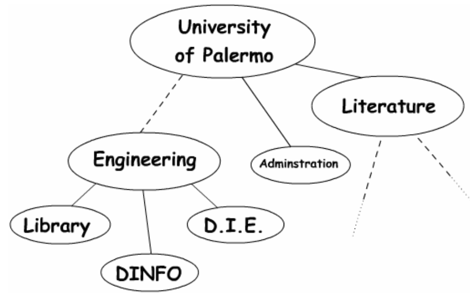

Figure 4. Network composition by area linking

4.2 Service Request

We implemented the interface between the user and the system agents by means of a MIDlet running on personal mobile devices supporting J2ME with MIDP 1.0 (or greater) APIs. The MIDlet is composed of a static part executed apart from the connection status, and a dynamic part which depends on the available services. The MIDlet first negotiates a mobile device connection to the network. This task is accomplished by means of the interaction between ConnectionEntities and UserEntities, which are implemented by mobile agents. Once the user is connected, the MIDlet contacts the UserEntity for service requests (See Figure 5). The UserEntity sends the appropriate MIDlet graphical interface by means of a XML protocol message, thus allowing system administrators to define dynamic graphical interfaces according to requested services. This way administrators have only to create XML files for appropriate GUI generation, thus avoiding MIDlet source rewriting for each needed GUI.

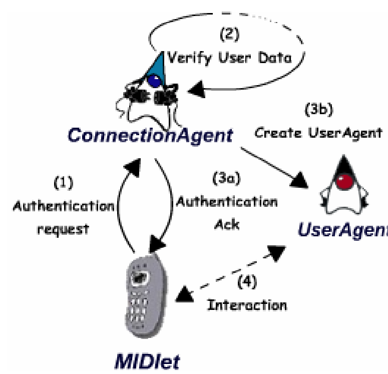

Figure 5. Mobile device - network connection steps

The request for a service involves two entities: a UserEntity and a ServiceEntity. The first one acts on behalf of users who need services, the second one supplies users with the required service. In more detail, when an user requires a service via the MIDlet on its mobile device, the UserEntity on the platform receives a packet with data related to the service required, and the name of the agent who can provide the service. This data is read from the `index.xml` file. Next the UserEntity asks the appropriate ServiceEntity for a service template, which will be used to correctly compose the service request. In fact, the service request format depends on the kind of available services and on the modes of service provided.

Once the template has been received, the UserEntity composes an appropriate form and sends it to the MIDlet. The users fills in the form fields with appropriate data, and sends it back to its UserEntity, which task is now to compile the XML node.

5. CASE STUDY IN A CAMPUS

We decided to implement the exam subscription service in order to evaluate the effectiveness of the proposed architecture. This kind of service is useful both for students and professors, thus allowing us to test reactions of both kind of users.

The service can be implemented by means of any wireless connection between mobile and fixed devices, and we decided to use Bluetooth for our tests. This choice allowed us to use SmartPhones as mobile devices, which are almost in every student's pocket. Furthermore, no fee had to be paid for the connection.

When an user comes within the coverage area of a Bluetooth AP, its device is discovered by the local Bluetooth service and the MIDlet is "pushed" on it (only if the user allowed the MIDlet installation). This task is accomplished by a simple Java application which loops a Bluetooth devices discovery sequence. Once a new device is discovered, the application establishes a Bluetooth connection with it and sends the MIDlet .jar file. Most mobile device's operating systems receive the file as a message attachment, and installation automatically begins upon user confirmation. All low-level interactions between fixed and mobile devices via Bluetooth take place by means of AveLink APIs from Atinav [9].

During our tests, we experienced some problem while pushing the MIDlet to some phone models, most likely due to differences among Bluetooth stack implementation. In particular, we noticed that Sony-Ericsson Z600 and Siemens S55 do not support the push mode, reporting a "unknown-file-format" when the message is received. On the other hand, most of Java-enabled Nokia models allow this interaction mode.

In an informal behavior analysis, we saw that students found this way for service provision more intuitive than a common web interface, since cellular phones are still more familiar than computers. Furthermore, cellular phones are in everyone's pocket, while computers connected to the internet are still more difficult to find for students within our campus.

As a confirmation of this informal result, we read an average value of 53 accesses per day from the system log files during the test period which lasted 25 days. This is a good result if compared with the number of students we have (200) and with the number of accesses we had via our website for the same task in the same period (39 per day average). Since there are no major differences from the access speed point of view with respect to the web-based access mode, we think that students appreciated the opportunity to access services at any time with no need to search for internet points and, above all, with no need to pay for internet access. Figure 6 shows the GUI corresponding to an exam subscription template.

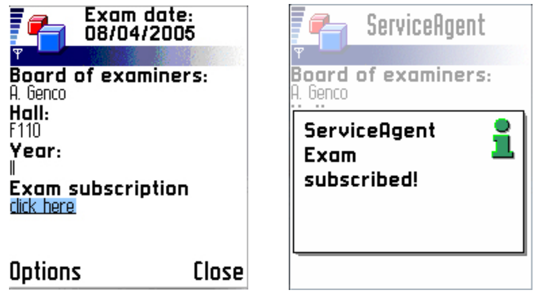

Figure 6. MIDlet GUI for exam subscription.

From the professor's point of view, since we defined a service template layout where few line of XML code are involved, it is straightforward to add, modify or delete exam information. Of course, professors take less advantage of the system than the students, but they found the system useful when some service-related data has to be changed rapidly. In particular, some of them changed the exam start time during an important meeting held a few minutes before the scheduled exam time.

We set up another service, based on the proposed architecture, in order to make a PDA or a SmartPhone suitable to operate as a remote controller on a remote display for applications that were not designed to run on such mobile devices. This is a good solution when multimedia contents or application outputs have to be shown to a quite large number of people. In our case, teachers with a presentation stored in their mobile devices, can first send the file to the remote display, and then interact with it. In particular, the implemented system allows users to control almost any kind of application running on a remote display.

We implemented a dual mode interaction for this service. The first one is devoted to be used on PDAs, so when an user is allowed to control a display, its PDA becomes a remote control for the display, showing volume controls or keyboard layout or cursor arrows, corresponding to the application running on the remote display. This interaction mode requires a specific application to be installed on the PDA.

The second interaction mode is to be used on SmartPhones, and it is based on the generation of an operational web-page, which shows commands to operate the remote application as web links. This interaction mode does not require any application to be installed on the SmartPhone.

Both solutions are based on a TCP/IP connection, such as Wi-Fi or GPRS, and of course both solutions provide users with an authentication phase.

6. CONCLUSIONS

This paper has shown how personal mobile devices can be conveniently used for service provision in augmented reality environments. A case study has been discussed which deals with resources and services being accessed in a campus, by means of a suitable model for hybrid entities exhibition in augmented reality. This has been accomplished by means of some protocols which allowed mobile devices and the hidden system to interact.

7. ACKNOWLEDGMENTS

This work has been partly supported by the CNR (National Research Council of Italy) and partly by the MIUR (Ministry of the Instruction, University and Research of Italy).

8. REFERENCES

[1] Agostaro F., Genco A., Sorce S. A Fuzzy Approach to Bluetooth Positioning. *WSEAS Transactions on Information Science and Applications*, (July 2004), Volume 1, issue 1, (ISSN:1790-0832), 393.

[2] Bellifemine F., Caire G., Poggi A., Rimassa G. JADE: a White Paper. *Exp*, Vol. 3, n..3, http://exp. telecomitalialab.com, 2003.

[3] Busetta, P.; Kuflik, T.; Merzi, M.; Rossi, S. Service Delivery in Smart Environments by Implicit Organizations. In *Proceedings of the IEEE MobiQuitous 2004 Conference*, (August 22-26, 2004), 356-363.

[4] En-Yi Chen; De-Gan Zhang; Yuan-Chun Shi; Guang-You Xu. Seamless Provisioning of Service in the Ubiquitous Computing Environment. In *Proceedings of the International Conference on Machine Learning and Cybernetics*, (November 2-5, 2003), Volume 3, 1904-1909.

[5] Foundation for Intelligent Physical Agents (FIPA). ACL Message Structure Specification. *Document number SC00061G*, date 2002/12/03, www.fipa.org.

[6] Foundation for Intelligent Physical Agents (FIPA). Agent Management Specification. *Document number SC00023J*, date 2002/12/03, www.fipa.org.

[7] Genco A. HAREM: the Hybrid Augmented Reality Exhibition Model. *WSEAS Transactions on Computers*, (January 2004), Volume 1, issue 3.

[8] Genco A., Sorce S., Scelfo G. Bluetooth Base Station Minimal Deployment for High Definition Positioning. In *Proceedings of the ACM-IEEE MobiQuitous 2005 Conference*, (July 17-21, 2005), 454-460.

[9] http://www.avelink.com/Bluetooth/

[10] Maekawa T., Uemukai T., Hara T., Nishio S. A java-based information browsing system in a remote display. In *Proceedings of the IEEE CEC Conference*, (July 6-9, 2004), 342-346.

[11] Uemukai, T., Hara, T., Tsukamoto, M., Nishio, S. A remote display environment: an integration of mobile and ubiquitous computing environments. In *Proceedings of the IEEE WCNC Conference*, (March 2002), Volume 2, 618-624.

[12] Wang, H.J.; Raman, B.; Chen-Nee Chuah; Biswas, R.; Gummadi, R.; Hohlt, B.; Xia Hong; Kiciman, E.; Zhuoqing Mao; Shih, J.S.; Subraimanian, L.; Zhno, B.Y.; Joseph, A.D.; Katz, R.H. ICEBERG: An Internet Core Network Architecture for Integrated Communications. *IEEE Personal Communications*, volume 7, issue 4, (August 2000), 10-19.