Middleware-Based Solution to Offer Mobile Presence Services

Victoria Beltran and Josep Paradells
Wireless Network Group – Telematics Department
Technical University of Catalonia
Mod. C3 Campus Nord, c\ Jordi Girona 1-3, 08034
Barcelona, Spain
{vbeltran, josep.paradells}@entel.upc.edu

ABSTRACT
Presence information is a subset of context that originated in Instant Messaging and Push to Talk applications. Presence information expresses all the determining factors behind communication between users, such as their availability, willingness, environment, and preferences, among other things. Applications can use presence to take intelligent decisions about the start and continuation of users' communications. Due to the growing trend towards integrating presence-aware applications into mobile networks, including cellular ones, software developers and users need efficient and scalable platforms upon which to deploy and use presence services. We propose a fully distributed platform to manage all user presence. It is a middleware based on a proxy server that retrieves and aggregates presence from different sources, applies high-level rules set by the user about presence privacy and communication adaptation, and implements strategies to reduce the amount of presence traffic sent on wireless links.

Categories and Subject Descriptors
D.2.11 [**Software Architectures**]: Domain-specific architectures

General Terms
Management, Documentation, Performance, Design.

Keywords
Context, Middleware, Presence, Proxy, Mobile.

1. INTRODUCTION
Presence information is a well-known concept on the Internet and is widely used by applications such as Instant Messaging and Push to Talk, in which the user can discover the willingness of other users in his buddy list to communicate with him, through the presence states of online, offline, busy or absent, among others. This basic understanding of presence is evolving towards a much more generic and flexible concept that includes all context that allows a user or application to adapt and control communications in a more efficient and personalized manner. Presence includes a wide range of information about a user, such as his localization, the activities that the user is doing at a specific time, ambient conditions, communications preferences and devices on which the user is available and even information expressed by abstract terms such as "intention" and "will". Presence is a powerful tool that offers a world of attractive possibilities for self-expression, letting our friends and contacts know how we are and seeing in an instant how they are. We are thereby able to choose the most suitable time to contact our buddies since we know when they are most available, and the condition in which we will find them. In this way we can avoid failed call attempts that sometimes end in voicemail, which in turn allows us to save money. Another significant motivating factor for using presence systems is our innate social curiosity that leads us to observe the activities and states of people relevant to us. Furthermore, presence can play its most useful role in the daily life of people whose working day is mainly taken up with meetings or appointments, since presence can help us to automate and organize our daily schedule. A presence-based application should only permit a user to communicate with another user by the services specified in the presence document of the latter. In addition, this communication should have the characteristics indicated in that presence document, such as user devices called or content types allowed. This property of presence systems allows us to control the way we communicate with other people by publishing appropriate presence information. For example, we can determine which communications to accept depending on the relevance of the requesters, redirect calls to secretaries or delegates when we are busy, know in real time where our employees are during a business trip and know exactly when collaborators have finished a meeting, writing a report or having lunch.

The concept of presence is understood as a type of context information, and there are many opinions about context and its use in pervasive computing. Context is defined as any information that can be used to characterize the situation of an entity. Although some definitions lead to an unclear differentiation between presence and context, presence should be understood as a specialized subset of context in two aspects: the object and the objective of the information. Context characterizes all relevant entities in any kind of interaction between a user and an application. This information helps applications to build all kind of intelligence that is aware of the user's environment. On the other hand, Presence is used to characterize entities that can affect the management of communications with a user. This information

Mobilware'08, February 12–15, 2008, Innsbruck, Austria.

allows applications to take intelligent decisions about the start, continuation and end of user communications.

Presence is the base from which a range of advanced applications that are innately associated with the mobility of users can be deployed. Mobile devices are personal and pervasive; the user always keeps them close, in a manner always-on, and stores personal information in them, such as a diary or favorite media. A seamless integration of presence-aware applications into mobile environments should be achieved. To this end, interoperable and scalable platforms are necessary to manage and exchange presence information in restricted networks. Today, there is an interoperability problem between presence systems despite presence information is focused on allowing personalized and intelligent communication between different users. Currently every solution uses proprietary implementations, so two users of different domains are not able to communicate with each other and the potential of presence is not exploited. Consequently, the IETF Working Group is looking to extend the SIP (Session Initiation Protocol) in order to provide a standard platform for instant messaging and presence. This platform is denoted by the acronym SIMPLE, or SIP for Instant Messaging and Presence Leveraging Extensions. The SIMPLE presence model is a subscription-based framework in which watchers send SUBSCRIBE messages to presence servers of interesting presentities, in order to keep up to date on the presence of those presentities. A presentity is an entity which has presence information associated and announces its presence by sending PUBLISH messages to its presence server. The presence server is the logical entity in charge of binding presentities to watchers and notifying watchers of proper presence changes via NOTIFY messages.

In addition to the interoperability problem carried over from the beginning of presence-enabled applications, the implementation of presence systems is being restrained by relevant unsolved issues. There are important open questions about user privacy. Where is the limit between the curiosity of watchers and the user privacy? Can users trust service providers to use their presence information appropriately?. These questions represent a barrier to the use and acceptance of presence systems, and users need flexible, context-aware and adaptable privacy models. Moreover, when presence applications are integrated into mobile environments, many more issues will have to be faced in the form of devices with high constraints, multiple access technologies and unreliable communication channels. Moreover, we have to take into account the amount of traffic generated by presence-enabled applications which send and receive periodic updates of subscriptions and user presence. This situation gets worse when mobile presence applications interact with Internet-designed applications since the latter do not take into account the volume of presence traffic. This excessive traffic entails two problems: overconsumption of bandwidth and battery power in user devices. Therefore, presence platforms adapted to the restrictions of mobile environments are needed in order to provide efficient and scalable mobile presence applications.

In this article the authors propose an innovative platform for managing users' presence that consists of a middleware as a personal proxy of the user. Our solution is fully distributed, network-independent and offers enough flexibility to add advanced functions based on presence. The rest of the article is structured as follows. We start with a brief overview of similar research work. Section 3 explains the operation of the middleware in more detail, followed by some design issues described in Section 4. In Section 5 we briefly describe how presence documents are built, and finally we give some conclusions.

2. RELATED WORK

Presence management is a beginner research field and it is difficult to find similar works that share our objectives. The entire related work only shares some features of our approach and then a deep comparison is not possible. However, following we describe some studies that deal with presence or context information.

Some studies about the integration of context and presence in wireless networks have focused on managing and storing this information in client mobile devices. Examples are [1] and [2]. The first offers a software platform installed on mobile devices to manage context, in which the device stores and controls all presence about buddies. In [2] a complex context-aware system is specified to control users' access networks and applications, in which buddies' presence is completely managed by user devices. Both examples are unaware of the consumption of network and device resources due to context management. Therefore, these solutions are not scalable as the number of buddies, or the amount of context associated with them increases. In our approach, the majority of intelligence is placed on the personal proxy which is in charge of reducing traffic load. The use of a proxy allows us to optimize network and device resources. The authors in [3] propose a generic platform for the provision and management of context in mobile environments in which the intelligence is placed in servers rather than in user devices. Our study shares some of the objectives of this solution, including simple user devices and user-controlled context management, reasoning, privacy, and exchange. However, it does not deal with presence information about the buddy list of the user or adapt his communications depending on his presence. The authors in [4] describe a platform for managing presence that relays on top of an OSA PAM (Presence and Availability Management) core and has ParlayX and SIMPLE interfaces. A PAM solution does not seem flexible, since it is based on CORBA, and the study also uses centralized servers, which can lead to scalability issues. Regarding cellular environments, the 3GPP has defined the integration of presence information into its specification IP Multimedia Core Network Subsystem (IMS) in UMTS. This system has been designed to converge data and voice over cellular networks and provide IP-based real-time services in UMTS. It is fully based on SIP protocol, and particularly on SIMPLE to manage users' presence and thereby offer cellular presence-enabled services. The standardization of presence applications into IMS is controlled by the OMA (Open Mobile Alliance) organization. The centralized platform that IMS provides to deploy operator-side presence systems has several drawbacks: scalability, billing, privacy and implementation problems. The presence traffic is managed by centralized mainframe servers that can become bottlenecks as the number of users grows, since presence applications generate large amounts of signaling traffic in updating users' presence. This amount of signaling traffic can make users reluctant to use cellular presence services since, for the moment, cellular operators do not offer affordable data rates to residential users. As far as privacy is concerned, users have to leave their presence information in operators' hands and they may not trust that this information,

which can contain personal data such as lists of activities, diary entries and engagements, will be handled correctly. In addition, OMA specifications to implement presence-enabled services in IMS have resulted in a large and complex set of documents that are hard to implement. Our solution is only comparable with a subset of the functionality of IMS related to presence functions. The previous problematic issues in IMS are solved in our solution implicitly thanks to the fully distributed nature of the proposed middleware. Each user has a personal proxy in his home or working place, not into centralized domains, and in addition, the proxy applies several techniques to reduce the amount of presence traffic over wireless links. This approach makes user growth and application of user-dependent privacy rules scalable and allows presence applications' rates cheaper. In [5] a variation of IMS is proposed in order to distribute presence hierarchically, thereby achieving more intelligent and personalized presence composition and privacy. Nevertheless this work only eases, but not resolve, some problematic issues in IMS. The authors in [6] show the need for interoperability between Internet-based and cellular applications. This study proposes a system based on subscriptions and managed by a central element that mediates between applications and users in cellular and Internet networks. Far from providing interoperability, this article describes a system to notify events between cellular and Internet networks. In order to reach interoperability between presence systems, these systems should implement the SIMPLE protocol which is being adopted as the presence standard by the research community. For this reason, our solution is fully SIMPLE standard. In [9] the authors state that presence information is the way of connecting Internet and cellular services and outlines an approach based on ontologies. This approach consists of a presence middleware that integrates and combines different user services in order to offer more advanced and personalized functions to users. The middleware combines rules, events and presence information to make decisions. We have a common idea with the authors: the building of intelligence from a set of presence attributes and rules to improve user services. However we do not share their purpose and architectures's technology.

3. PRESENCE MIDDLEWARE

In the present study we propose a fully distributed solution for deploying presence-aware applications that consists in a middleware acting as a personal proxy in the home or office of the user. This approach implements the IETF SIP/SIMPLE presence model and its main task is the intelligent search and management of presence to allow applications to adapt their behavior based on this information. In addition, this model gives enough flexibility to apply fine-grained rules about user presence privacy, communication adaptation and traffic optimization strategies. Although some middleware functionality is placed in user devices, the majority of functions are installed on a computer, a proxy, connected to Internet by a wired network, which functions as the intermediary point for any presence exchange with the user. This proxy can run on any computer owned by a user or can be a box offered by an operator or company which will be responsible for the majority of its management. In any case the proxy provides an API to give the user the ability to set his static- or context-dependent preferences with regard to privacy or communications. This approach permits us to offer a single point in charge of collecting, storing, managing and publishing user presence, so users do not have to carry out these functions in any of their devices. This approach allows more efficient and lighter user devices and device-independent access to user presence. Presence information is available even when the user is not connected to any device and then this information is always-on. What is more, the proxy aggregates user presence from different sources in a consistent way, that is, the proxy creates a correct single view of user presence removing any contradictory, redundant or stale information.

Regarding presence-based communications, the proxy is the intermediary point through which the user is contacted, so external entities never know the user's physical address. Thus, security is increased by allowing users to keep their localization secret. The described presence middleware is a scalable solution by nature, close and personal to the user, which provides interoperability between presence services running on different networks. Although this middleware can work in any scenario, it resolves problems of other centralized solutions in mobile environments and enhances the performance of wireless links. In mobile networks, user devices usually have scarce resources so processing costs in these devices have to be taken into account when presence-based applications are provided. What is more, presence information is encoded as XML documents that are not appropriate for mobile networks due to both the processing requirements needed to analyze these documents on user devices and the limited capacity of wireless communication channels. These matters have been considered in the design of the presence middleware, which focuses on reducing the complexity of user devices and utilizing wireless links efficiently, which is achieved thanks to the collaboration between the client and server parts of the middleware. This collaboration allows us to implement non-standard solutions between the user device and the proxy in order to obtain higher performance than the fully standard solutions. The middleware decreases, as far as possible, the amount of presence information exchanged with the user, but it also has enough intelligence to use different optimization techniques depending on the user's context. Due to the fact that some techniques require extra processing on user devices, a balance between benefits and costs must be found when optimizations are applied. For this reason, the context of user devices' characteristics is very useful when deciding which techniques to apply. This paper focuses on implementation issues but does not deal with optimization techniques. For further information on some of these techniques, see [7].

4. ARCHITECTURE DESIGN

The architecture of the proposed presence middleware is divided into two logical layers: the Management layer and the SIP/SIMPLE layer. The former contains the intelligence needed to process and manage presence information and the latter is responsible for receiving and sending messages related to SIP and SIMPLE protocols. The following points describe these layers briefly and figure 1 shows the structure of each one into both client and server sides of the middleware. The client middleware, called CPM (Client Presence Middleware), is far simpler than the server part, since mobile devices usually have limited processing and memory resources. It offers an API (the two-toned rhombus in the figure) to all applications of client devices in order to provide the intelligent functions about presence of the server middleware in a transparent way. The server middleware, called Server

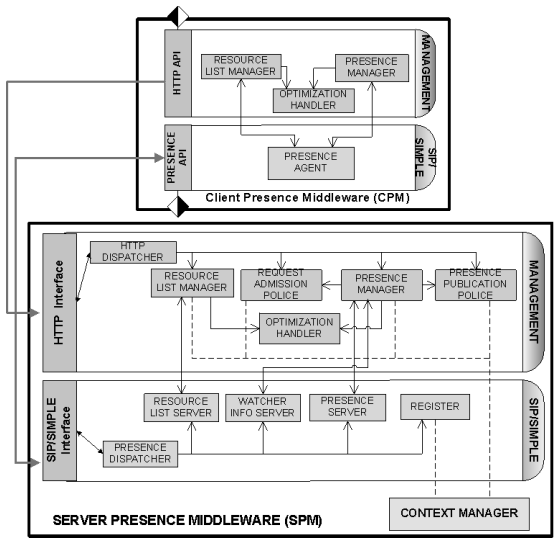

Figure 1. Communication between the modules of the Presence Middleware.

Presence Middleware (SPM), which functions as the proxy, stores user presence in a context repository called Context Manager.

The presence information processed by the middleware can be classified in different groups: Personal Information, Services, Resource List and User Presence Rules. The first group contains information that is closely related to the user and may include mood, activities, willingness to communicate, ambient conditions, profiles, personal addresses and localization, among others. Services group includes information about available services by which the user is contactable, and devices that support those services. For example, information about a service may include content types accepted by the service and hardware and software characteristics of devices where the service is available. The Resource List includes all presence of the entities which the user is watching. To simplify we assume that an entity is a person, although it could be a software entity. For this reason, we name a resource as buddy and the resource list as buddy list. The last group contains rules that allow users to build their model about data privacy and communications with other users, and these are grouped depending on their purpose. The Request Admission Rules (RA rules) indicate the communication types that the user is willing to establish with other users. The Presence Publication Rules (PP rules) establish the privacy level when user presence is published and finally the White and Black List Rules accept or reject, respectively, an unknown (non-authorized) entity that is requesting to watch the user's presence.

The presence middleware has been developed on the Java Platform. The SPM has been programmed using Java Platform Standard Edition (J2SE) and it has been developed on top of a SIPMethod Application Server that is JSR-116 (SIP Servlet API) compliant SIP Application Server runtime engine. Currently, only a CPM over mobile phones has been developed, and for which purpose we used Java Platform Micro Edition (J2ME) with CLCD (Connected Limited Device Configuration) and MIDP (Mobile Information Device Profile). A Java API, JSR 180 (SIP API for J2ME) has been used to implement the SIP communication between CMP and SPM.

4.1 Management Layer

On the server side, the *HTTP Dispatcher* receives HTTP requests that contain user information such as configuration data, preferences or rules. This information is put into the user device (usually by means of a Graphical Interface) and is sent to this module, which communicates with other modules within the same layer depending on the type of information received. The *Presence Manager* processes and aggregates the presence published by the user, and also generates presence documents that are sent to the user's watchers. In order to create legal presence documents, it communicates with the *Request Admission Police* and the *Presence Publication Police* modules. These two modules contain a set of RA and PP rules respectively. These rules decide by which services the user is willing to communicate with other users and the privacy level of presence documents. The presence information associated with the user's buddy list is managed by the *Resource List Manager* module, which is responsible for processing and aggregating presence notifications from the buddies. In addition, it creates presence documents about the buddy list and orders the *Resource List Server* module to send these to the user. Finally, the *Optimization Handler* contains the intelligence needed to decide the most suitable optimizations depending on the circumstances of the user, device, communication channel, preferences and statistics.

On the client side the complexity of the middleware is much lower. The Presence Manager is in charge of controlling and updating presence information about the user so that, when the user makes a change, it is responsible for advising *Presence Agent* to publish the presence changes to the SPM. The Resource List Manager only stores presence about the user's buddy list and offers it to applications. Last of all, the operation of the Optimization Handler is very basic and mainly consists of configuration information about possible optimization strategies of the presence traffic.

4.2 SIP/SIMPLE Layer

The single module on the client side is the *Presence Agent* which carries out basic tasks related to the storage of user and buddy list presence and the registration of user localization. It registers the physical address of the user device in the SPM by REGISTER messages. Every time user presence has to be published it sends a PUBLISH message to the SPM, and it also receives NOTIFY messages sent by the SPM containing presence about the user's buddies.

On the server side, the *Presence Dispatcher* receives SIP/SIMPLE requests sent by the CPM and resends them to proper modules. The *Register* module saves the physical localization of the user device which is set in the REGISTER messages sent by the CPM to the SPM. Thanks to this register, other modules in the SPM can send messages to the user's devices and external applications can establish sessions with the user. The *Resource List Server* is responsible for sending and receiving all SIMPLE messages related to the management and maintenance of the user's buddy list. Its main tasks are the subscription to each buddy via SUBSCRIBE messages and the reception of NOTIFY messages that update the buddies' presence. In addition, this module keeps the subscription for each buddy alive, which involves sending periodic SUBSCRIBE messages to avoid subscriptions expiring. The *Presence Server* module receives PUBLISH messages sent by

the CPM that contain user presence and it also sends NOTIFY messages to the user's watchers in order to notify them of changes in the user presence. In addition, this module controls the state of the subscriptions associated with watchers. It has to update the state of each subscription depending on the SUBSCRIBE messages received and the expiration time for the subscription. Finally, the *Watcher Info Server* notifies users of the arrival of a request to watch their presence from an unknown entity. This module waits for the decision of the user before permitting the unknown entity to watch user presence (or not). This module is called by the Presence Manager when the latter receives a SUBSCRIBE message from an unknown watcher. Then the Presence Manager checks if this watcher is in the black or white list, which would allow it to take a default decision. If the watcher is in neither of the lists, the Watcher Info Server has to send a NOTIFY message asking the user for authorization, and later the user will make his or her decision known via a PUBLISH message.

5. PRESENCE FILTERING

Whenever the user changes his presence, the CPM sends a PUBLISH message to the SPM which then notifies each watcher of the changes done via a NOTIFY message that contains a suitable presence document. In the following two points, we are going to see what type of information this presence document contains and how this information is encoded.

5.1 Presence Documents

The presence model of [8] includes three components: Person, Service and Device. *Person* represents the user with which the presence is associated and includes personal information such as state, address, activities, mood, localization, features of the user, name, address, feelings, etc. *Service* means a service that is a point from which to communicate with the user. *Device* means a device, a physical environment, where a service is running. A user can have zero, one or more services at his disposal, through which he can establish communication with other users, and each service is associated with the user devices where it is running. According to this presence model, the SMP generates a XML presence document divided in three different parts. First, all personal information about the user and his environment is collected in a single <person> tag that follows the RPID [13] and CIPID [11] specifications. For example, RPID can express user availability by states (offline, bussy, etc.), activities that the user is doing at some time and place type where the user is. The second part of the document is composed by a set of <tuple> tags, each of them representing a service available to communicate with the user which is encoded by PIDF [10], RPID and PRESCAPS [12] specifications. Some examples of service properties are state, contact address, identifiers of devices where the service is available, and capabilities of the service (such as media type or SIP event packages accepted). Finally, the third part of the presence document is the description of the devices that make possible some service. Devices are represented by <device> tags and each of them has to be referred at least by one <tuple> tag.

5.2 Presence Filtering

The user is able to set policies about presence privacy and communication preferences via HTTP protocol. These policies are

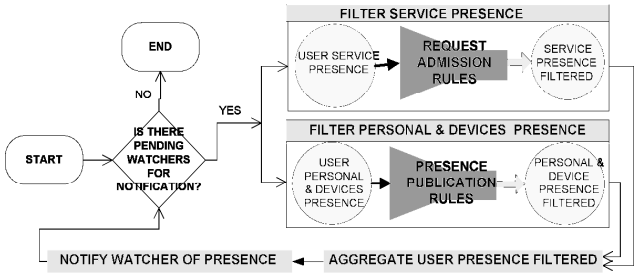

Figure 2. Flowchart to notify watchers of presence documents

called RA and PP Rules, are managed by RAP and PPP modules, and are useful whenever the SPM decides to notify watchers of user presence changes. The presence documents sent to watchers must obey these rules. In figure 2 the process of notification and presence filtering is outlined. Following a more detailed description of RA and PP rules is given.

The RA rules restrict the communication types that the user can establish with other users. Internally, these rules point out which presence about services is going to be included in the presence documents sent to watchers. Watchers communicate with a user by the services included in his document, and in this way, RA rules force watchers to communicate with the user by certain ways. The determining parameters of these rules are the watchers and the user's presence. Some examples are "I don't accept instant messages from team mates", "I only accept video when I have *leisure* state", and "I accept only audio from my boss when I am connected on *PDA7865* device". Presence publication allows us to force the rest of users to communicate with us by certain ways. In figure 3 we can see a clarifying example where a user, Alfred, has two devices, a personal phone identified by nokia6280 and a work PDA identified by acerc510. In the top of the figure the presence tree of Alfred is shown. This tree indicates that his two devices have instant messaging (IM) and video services. However, Alfred does not want to communicate with all persons of his buddy list by the same way. When he is working, he is pleased with any type of call from his wife, Theresa, but always in his personal phone. Regarding school friends, he is only willing to accept instant messages sent to his personal phone. In the case of his work mates they are only allowed to send instant messaging and video to his acerc510 device. All these preferences would be established in the presence proxy of Alfred by the following rules: "I accept IM and video in *nokia6280* sent by Theresa when i have *working* state", "I accept IM in *nokia6280* sent by any buddy in my group *School Friend* when i have *working* state", "I accept IM and video in *acerc510* sent by any buddy of my group *Work* when i have *working* state". In the bottom of the figure we can see the Alfred's presence tree received by Theresa, school friends and work mates.

The Presence Publication rules allow users to indicate the privacy level of personal information in presence documents depending on their presence and watchers. Some examples are: "Do not publish all my personal presence to my work mates", "Do not publish my vCard to my partners" and "Do not publish my personal presence to any buddy when I am connected to *PDA987* and my state is *working*".

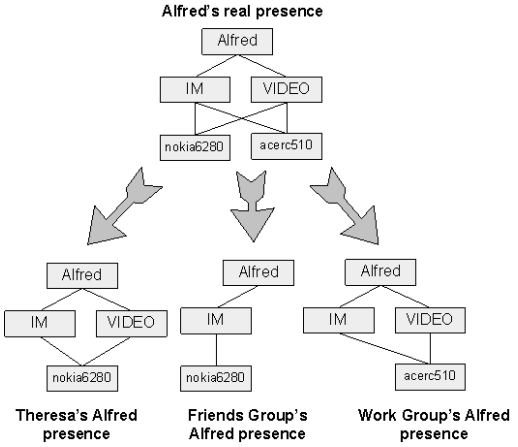

Figure 3. Flowchart to notify watchers of presence documents

6. CONCLUSION

We have argued for the current and future importance of presence-aware services in providing advanced functions for the efficient, personalized and context-aware adaptation of users' communications. In addition, we have presented a review of the most relevant issues about the management of presence and context information in mobile environments. Our contribution is a fully distributed platform for controlling all user presence in a scalable and interoperable way. This solution is a middleware that collects presence about users and their buddy lists from different sources and adapts its behavior depending on high-level fine-grained user rules about privacy and communications. A relevant feature when the proposed platform works on mobile environments is that it is able to apply strategies to reduce the amount of presence traffic sent on wireless links. These strategies vary dynamically depending on the available resources of the communication channel and user device, and the preferences set by the user. In order to improve the overall performance we are currently investigating the impact of this platform on mobile devices with limited resources, and new communication strategies between the client side and the server side of the middleware.

7. ACKNOWLEDGMENTS

This work was supported in part by the Spanish Government through CICYT project [TIC2006-04504] and a grant from the Ministerio de Educación y Ciencia [FPU AP2006-02846].

8. REFERENCES

[1] Raento, M., Oulasvirta, A., Petit, R., Toivonen, H. 2005. ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications. IEEE Pervasive Computing, vol. 4, Issue 2, pp. 51-59, April 2005.

[2] Inoue, M., Mahmud, K., Murakami, H., Hasegawa, M., Morikawa, H. 2005. Context-Based Network and Application Management on Seamless Networking Platform. Wireless Personal Communications, vol. 35, Issue 1-2 (October 2005), pp. 53-70.

[3] Sinderen, M.J., Halteren, A.T., Wegdam, M., Meeuwissen, H.B, Eertink, E.H. 2006. Supporting Context-Aware Mobile Applications: An Infraestructure Approach. IEEE Communications Magazine, September 2006, pp: 96-104.

[4] Wegscheider, F., Bessler, S., Gruber, G. 2005. Interworking of Presence Protocols and Service Interfaces. In Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005). IEEE International Conference. vol. 4. pp. 45-52.

[5] Brok, J., Kumar, B., Meeuwissen, E, Batteram, H.J. 2006. Enabling new services by exploiting presence and context information in IMS. Bell Labs Tech. J., vol. 10, Issue 4, March 2006, pp. 83-100.

[6] Gurbani, V.K, Sun, X. 2005. A Systematic Approach for Closer and Integration of Cellular and Internet Services. IEEE Network, February 2005, vol. 19, Issue 1, pp. 26-32.

[7] Beltran, V., Sanchez-Loro, X., Paradells, J., Casademont, J. 2007. Optimization of Presence Enabled Services over Cellular Networks Based on a Personal Proxy. In IASTED Conf. on Internet and Multimedia Systems and Applications (EuroIMSA), March 2007, pp. 75-81.

[8] Rosenberg, J. 2006. A Data Model for Presence. RFC 4479. Internet Engineering Task Force, July 2006.

[9] Shen, Q., Liao, Q.S. 2005. Presence: the Glue of Cellular and Internet Services. In IEEE International Symposium on Communications and Information Technology, ISCIT 2005, pp. 784-787.

[10] Sugano, H., Fujimoto, S., Klyne, G., Bateman, A., Carr, W., Peterson, J. 2004. Presence Information Data Format. RFC 3863. Internet Engineering Task Force, August 2004.

[11] Schulzrinne, H. 2006. CIPID: Contact Information for the Presence Information Data Format. RFC 4482. Internet Engineering Task Force, July 2006.

[12] Lonnfors, M., Kiss, K. 2007. Session Initiation Protocol (SIP) User Agent Capability Extension to Presence Information Data Format (PIDF). Internet Draft, draft-ietf-simpele-prescaps-ext-08. Internet Engineering Task Force, Sept 2007.

[13] Schulzrinne, H., Gurbani, U., Kyzivat, P., Rosenberg, J. 2006. RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF). RFC 4480. Internet Engineering Task Force, July 2006.

[14] Dey, A. K., Abowd, G.D. 1999, Towards a Better Understanding of Context and Context-Awareness. Technical Report GIT-GVU-99-22, College of Computing, Georgia Institute of Technology, Atlanta GA USA, 1999.