Building Infrastructure Support for Ubiquitous Context-Aware Systems

Wei Li, Martin Jonsson, Fredrik Kilander, and Carl Gustaf Jansson

Department of Computer and Systems Sciences,
Swedish Royal Institute of Technology,
Forum 100, 164 40 KISTA, Sweden
{liwei, martinj, fk, calle}@dsv.su.se

Abstract. Many context-aware systems have been demonstrated in lab environments; however, due to some difficulties such as the scalability and privacy issues, they are not yet practical for deployment on a large scale. This paper addresses these two issues with particular interest in user's privacy protection and spontaneous system association. A person-centric service infrastructure is proposed together with a context-aware call forwarding system constructed as a proof-of-concept prototype based on the Session Initiation Protocol (SIP).

1 Introduction and Motivation

During the last decade, numerous efforts in the Ubiquitous Computing arena attempted to realize Mark Weiser's vision of the 21st century computers – to enable computers "weave themselves into the fabric of everyday life" [1]. The purpose of making computers invisible is to diminish the unnecessary distraction introduced by computers so that users can concentrate on the higher level tasks they are currently involved in. Meanwhile, computers should assist *behind the scene* [2] without the occupation of users' attention.

Today, with the rapid development in mobile and distributed computing, especially the flourish of wireless communication and resource discovery technologies, various computing resources, in terms of *services*, are becoming transparently available everywhere. In the meantime, sensor technologies have been widely adopted to provide rich information for facilitating user's interaction. In the midst of the trend that services and sensors become pervasive, the goal of ubiquitous computing has been approached by many researchers with their prototypes in different testbed lab environments [2, 3, 4].

One common issue, however, among these existing systems is scalability. Due to the fact that each of these systems uses individual data expression and communication means, they are not interoperable with each other. As none of them could be dominant in the field, a mobile user roaming from one system to another normally have to conduct a significant number of interactions, typically involving in new software installation and configuration, to be able to obtain an access to the new surroundings. These extra interaction efforts that often result in a deviation of the user's attention

J. Cao et al. (Eds.): ISPA 2004, LNCS 3358, pp. 509–518, 2004.

are in many cases unappreciated or not even *unaffordable* by mobile users who are inherently confronted with more interaction and communication constraints during the move.

Another serious issue is user's privacy in ubiquitous computing environments. With the advance of sensor technology, ever more information can be captured from both the environment and the user. This might give more possibility to simplify user's interaction with computer systems, but also introduce a strong risk towards user's privacy intrusions. The exposure of fractional personal information occasionally slowly over time may still result in a substantial privacy loss after accumulation. Therefore it is becoming a critical issue in the ubiquitous computing field how to protect the user information acquired by the systems from being misused or disclosed to other undesired parties.

By identifying these two crucial issues, along with other important concerns, we claim today's ubiquitous computing environment should have the following characteristics:

Open and Standard: A ubiquitous computing system should be based on open data formats over standard communication protocols to achieve better scalability.

Low Demands on User Devices: To become widely available, a ubiquitous computing system should not demand too much of user's terminal devices, e.g., their computation and battery power etc. It should instead accommodate the most widely used/affordable off-the-shelf devices such as today's mobile phones and Personal Digital Assistants (PDAs).

Exploiting Infrastructure Support: It is of great significance to alleviate the resource-limited user terminal devices by migrating computation into the system infrastructure, exploiting the vast resources available on the local and remote computers.

Context-Awareness Support: It is impractical or even *impossible* for a mobile user to manually select services from a big amount and maintain the interactions with them. Hence a common support is needed to enable the system make decisions on *behalf* of the users based on information regarding their current situation (or *context*).

Privacy-Protected: Due to the risk of privacy leak, the major challenge is how to ensure the users, participating in a large-scale ubiquitous context-aware system, which is likely to cross multiple organizations and administrations, with an acceptable degree of control of their personal information.

To meet the characteristics listed above, we propose a person-centric system architecture which we believe suitable for a large-scale deployment. Within this architecture, we have addressed context processing and transportation difficulties with particular interest in user's privacy protection. We have also tackled the dynamic system association (also referred to as *bootstrapping*) problem with special concerns for mobile users. We have developed a set of software components and services, for the purpose of this paper, the toolkit is called ***S***ervice ***T***oolkit for ***A***daptive ***C***ontext-Aware ***S***ystems – *STACS*, to simplify the development of context-aware services and the construction of ubiquitous computing systems.

In section 2, we first introduce our system architecture in general, and then give more details on how to enhance security and privacy protection by enabling pseudonym-based communication. In section 3, we elaborate our service toolkit – a set of software components and services, together with a prototype context aware call forwarding system to illustrate how STACS can be used to build a ubiquitous computing system in a simple manner. Then in section 4, we discuss some related work with comparisons. And finally in section 5, we draw some conclusions based on our experience with some words of the future work.

2 System Architecture

We designed a *person*-centric software architecture in our ACAS project [5], to provide *affordable* and *adaptive* support for facilitating mobile users' interactions with services in ubiquitous computing environment. In this architecture, we assume each user has a persistent digital representative, called *Personal Server* (PS). This Personal Server is connected to the Internet, and most likely resides on the user's *Home* network. Each user is equipped with a mobile device, an off-the-shelf mobile phone or PDA. As the users move around, across *different* ubiquitous computing environments, they will be associated (and *de*-associated) with those systems spontaneously and the relevant context information (location and available services) will be reported to their individual home Personal Servers.

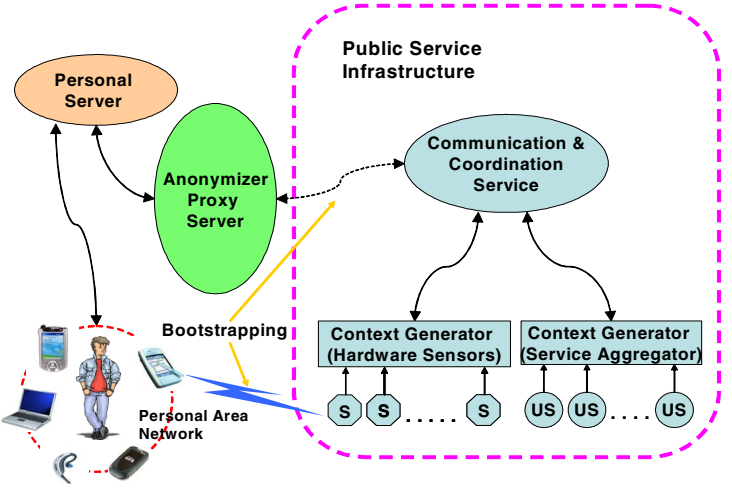


Fig. 1. System Architecture (Context Flow) US: end-User Service and S: Sensor

Two non-mutually exclusive interaction models are possible in such a setting: (1) users (with their devices) interact with the local systems directly or, (2) the PS acts as a remote intermediary between the user and local systems. We prefer the later model due to its simplicity (towards the user) that a single event to the PS is necessary for

associating (bootstrapping) user's interaction with the local environment. Once the PS acknowledges the local system, it may conduct more interactions *on behalf of* the user, e.g., to employ the services in that location. Hence the computation and communication efforts required to maintain secure interaction can be reduced greatly from user's mobile devices. This complies well with what we claimed for a large-scale ubiquitous computing system: **Low demands on user devices** and **Exploiting infrastructure support**. In contrast, the former model is more subject to difficulties such as increased battery power drain and vulnerability to network-based attacks, although we do not deny the necessity and flexibility of direct interactions, especially when the PS is incapable (or unreachable) to handle the interaction with local resources. However, this method does not provide sufficient means to protect the user's privacy since the PS is exposed to the local infrastructure system, which results in the presence of our *Anonimizer Proxy Server*. As shown in Figure 1, our infrastructure consists of two sub-systems: the Public Service Infrastructure and the user's Personal Server.

Public Service Infrastructure (PSI): each PSI is an instance of a ubiquitous computing environment. A PSI may be confined to a room, a vehicle, an organization or any other natural or abstract boundary. Although PSIs are orthogonal to geo-location spaces it is likely to be conceptually convenient to create mappings between them. Within the PSI, context data is produced by *context generators* which are attached to *hardware* sensors that measure physical properties (e.g., temperature), or *software* sensors that measure computing properties (e.g., available services) of a PSI.

The PSI representative is called the *Communication and Coordination Service* (CCS), which manifests a PSI to the Internet as an addressable entity and would communicate with Personal Servers for exchanging context information. A CCS receives context data produced by the context generators from the same container PSI. When context indicates that a user having a home Personal Server is present within the PSI, the CCS adds the user's PS address as a temporary resource of context and services.

Interaction Bootstrapping: There are two complimentary ways to associate user interaction with a PSI: (1) the user informs her PS of the current PSI, or (2) the PSI actively contacts the user's PS if it gets her PS address when detecting the user. In our design, these steps are augmented by sensor technologies to support spontaneous bootstrapping.

Anonymizer Proxy Server (APS): A user who does not wish to be tracked by PSIs should use an APS to protect her true identity (superior than turning off some device capabilities). Instead of providing a PSI with the real address to the user's Personal Server (which is assumed to be persistent), the user offers a temporary token, previously negotiated with the APS. The token allows a PSI to communicate with the user's Personal Server via the APS until the token expires or is withdrawn. Disclosure of additional information is then determined by the user's Personal Server.

Personal Server (PS): As the persistent on-line representative of a user, the Personal Server contains a set of service components including the individual's context reposi-

tory and a personal context manager (Figure 2). The context repository comprises different sets of information: the user device information, such as those in the Personal Area Network (PAN); personal contacts, calendar, and preferences (policies for sharing personal context); as well as the dynamic contextual data reported from the user's current PSI or personal devices.

The *Context Manager* handles the context reports and requests received through the *Personal Context Portal*, which gives an interface for external access to the user's context. It could also validate the reported context data before putting it into the *Context Repository*; or authenticate the context requester against the user's access policies. The external communication and interaction (with the owning user) requests arrive through the *Communication Portal* offering different access means, e.g., a set of Web Services over SOAP/HTTP. These communication requests are handled by different context-aware applications which can interface with the Personal Server system. Each application utilizes a relevant set of context information through the Context Manager to determine its application-specific logic and present the result to the user in different adaptive ways.

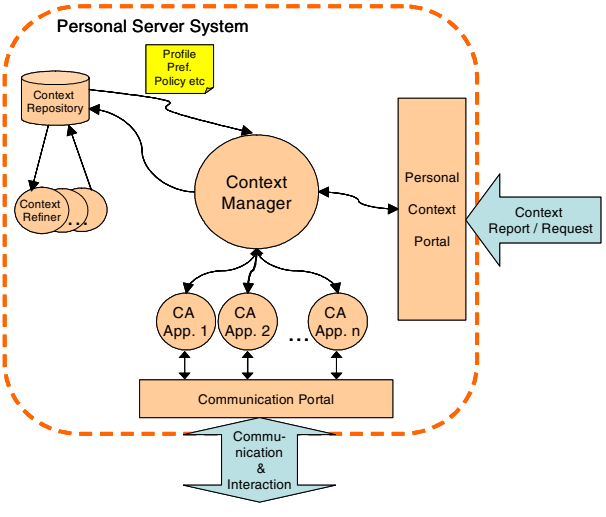


Fig. 2. Personal Server Inner Structure CA: Context-Aware

We also introduce *context refiners* which are used to produce abstract higher-level context information based on the data in the repository. A context refiner consists of a set of rules combined with a rule engine. The rules are compiled from documents, normally one set for each context-aware application. Whenever new context data arrive in the PS, the rules are applied by the reasoning engine to filter out irrelevant context information and insert new inferences. The result set derived is posted back into the context repository as available context data. For instance, a context refiner could generate an 'in-meeting' event based on the information of a user's current location and the nearby persons. This is implemented in our prototype (see Section 3.3).

2.1 Security and Privacy Protection

User preference and policy together with data encryption technologies have been commonly used to provide security and individual privacy protection in existing ubiquitous computing systems like [6, 7, 8, 9].

Different from those methods, we leverage the *pseudonym* communication mechanism (which was mentioned briefly with the introduction of Anonymizer Proxy Server (APS)), which grants access to users' context information without exposing their real identities. We thus propose a secure context exchange approach: A set of user pseudonyms (in form of some neutral tokens generated by randomization) are registered with an authentication authority in advance, and each pseudonym will be used as a subject reference to which a context requester (PSI or another user) will ask the authority for access. This request may trigger an authentication process and only the ones who pass the authentication will be granted to further contact the APS. In a simplified case, an APS can act as an authentication authority simultaneously since they are based on similar principles. By using pseudonyms, a context requester (PSI or other people) would not know and therefore could not retain personally identifiable information, but when needed can deal with abuse through the help of the authority or APS. By using multiple and replaceable pseudonym, issued access grants can be audited, revoked and blocked. Finally, with an authentication authority, system integrity is improved by preventing fraudulent access. However, a pseudonym only makes sense when the number of users is *reasonably* large, so that the user can hide in the crowd behind the APS. Also as most computer communication relies on low-level static identification such as IP and network card MAC addresses, a user is always at risk of being identified or tracked through the local communication [10].

3 Implementation

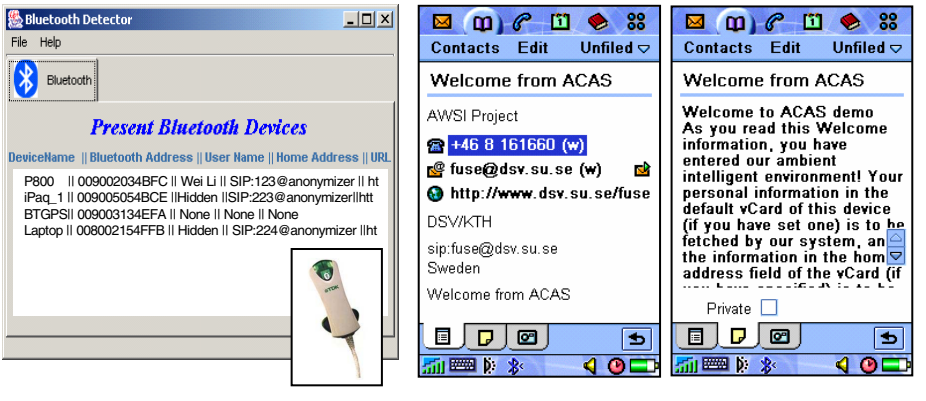
We have developed a set of software components as well as some services built on top of these components, which all together form our prototype service toolkit – STACS. We believe that by using the STACS, the construction of scalable ubiquitous context-aware systems conforming to the characteristics we claimed can be simplified and accelerated.

3.1 Communication Components

The context distributing in our architecture is based on a Subscribe/Notify/Publish mechanism. The Session Initiation Protocol (SIP) [11] is used as the underlying communication protocol, mainly because of the agile tolerance SIP defines for handling communication sessions over unreliable networks, as well as its openness and wide acceptance. Our implementation conforms to the SIP Presence Framework [12]: A SIP *Presence User Agent* (context producer) publishes sensed (low-level) context

data to a SIP *Presence Agent* (context provider) which, after some processing of the data, will notify the SIP *Presence Watchers* (context consumers) who have registered interest about the generated (high-level) context results. These three SIP Presence entities have been implemented as the primary software communication components in our STACS. By making a diverse use of these SIP components (as *context sockets*), various system entities can be plugged into each other to construct a dynamic context information network.

The Context Manager in a user's home Personal Server system employs an internal Presence Watcher to subscribe and receive context information about a PSI through its Communication and Coordination Service (CCS), which is implemented based on a Presence Agent. This subscription goes through an APS which is also implemented as a Presence Agent. The Context Generators work as PUAs, publishing derived context information to the infrastructure CCS, and some of them (desired by some users' Context Manager) will be further delivered to their Personal Server systems as context notifications. The Context Manager in a Personal Server may also employ a Presence Agent for sharing personal context with PSIs or other users. There are other ways of using these components for delivering context data: e.g., a Personal Server may hire a Presence Watcher to subscribe to user's mobile devices for acquiring context data from a Presence Agent running on that device, or the user may start a PUA on the device to publish context to her Personal Server. The combination use of these components gives great flexibility for context acquisition, processing and distribution among entities across the network, or within the Personal Server and PSI systems. The superior scheme can be determined according to the requirements in concrete scenarios.



(a) (b) (c)

Fig. 3. (a) Bluetooth Detector using a TDK USB Bluetooth dongle. (b) The system entry addresses (sip:fuse@dsv.su.se) and (c) the Note part of a system Welcome vCard received by user's mobile phone (SonyEricsson P800)

3.3 Prototype System

A Context Aware Call Forwarding application based on our person-centric system architecture with the use of STACS has been developed as a prototype example. This application monitors a user's context changes, and will set the call forwarding when a meeting situation is determined depending on two facts: if the user is in a meeting room and not being alone. The detailed communication flow is illustrated in Figure 4.

A desktop PC in our meeting room, acting as an Infrastructure (PSI) server, runs a Bluetooth detection service to detect and retrieve a vCard file (step 2.) from a user's mobile device. It will then report the location (meeting room) to the pseudonym address (placed in the retrieved vCard) referred to as the user's Personal Server System. This context report will first arrive (3.(a)) at the corresponding Anonymous Proxy Server and then be forwarded (3.(b)) to the Context Manager (with an internal Context Repository) within the user's Personal Server system. The location report will be further delivered (step 4.) to the Meeting Monitor service (a context refiner) as a context notification, which will trigger it to infer (step 5.) the meeting status. Once approved, the Meeting Monitor will activate (step 6.) the Call Policy Generator to produce a call forwarding script (in Call Processing Language [15]) according to the user's preference, and finally upload it to the user's Personal Communication Server for enforcement (step 7.). The Personal Communication Server used is Vovida Vocal [16], an open-source SIP proxy server with support of call processing scripts. Xten X-Pro 1.0 [17] has been tested as SIP softphones on HP iPAQ 5550.

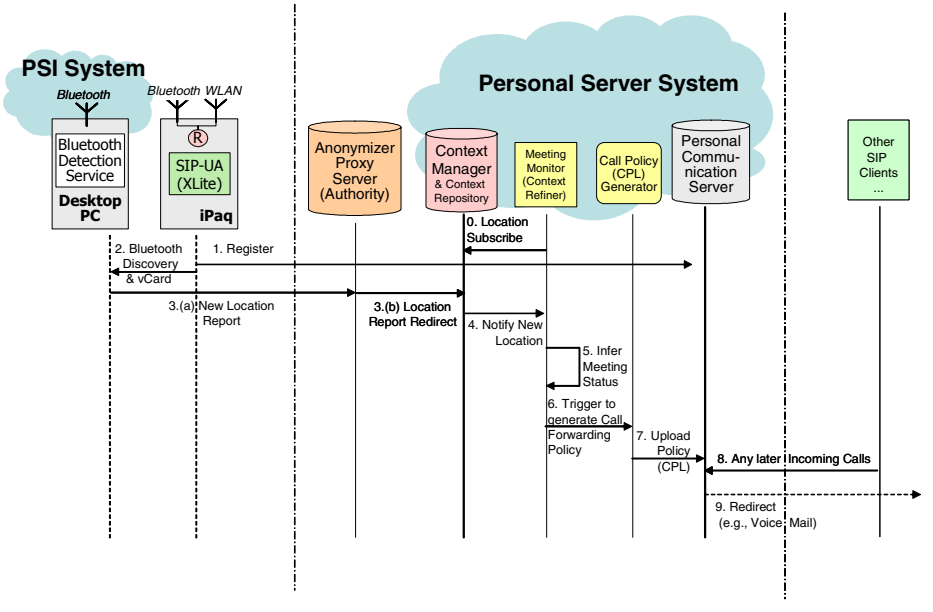


Fig. 4. Context-Aware Call Forwarding System

4 Related Work

Due to the large number of research works to date in ubiquitous and context-aware computing area, we will only address some of them which are most relevant to our work. K. El-khatib et al. [18] used Personal Agent (also implemented on SIP) to determine how to render mobile user's incoming calls in a ubiquitous computing environment with support for better performance and interaction means according to a user's profile and the available services. Stefan Beger and Henning Schulzrinne et al. have elaborated comprehensively in a recent paper [19] on how to construct ubiquitous computing system using SIP together with many other standard protocols. We agree with them that a global-scale ubiquitous computing system should be divided into different domains, and through the SIP servers in those domains, the user can utilize the rich resources in the visited domains. However, except for many similarities mainly because of the use of common technologies and standard protocols such as Bluetooth and SIP, there are clear differences to distinguish our work: firstly, we have described how the local infrastructure instead of user's mobile device, can deliver context data back to the home Personal Server system without exposing user's identifiable information; secondly we introduced a context refiner concept located at the user's home system to infer high-level context information. In additional there are many differences in design details, for example our Bluetooth detection service does not need any action by the user, while theirs needs the user's device to discover the Bluetooth access point to generate location information etc.

5 Conclusion

To achieve a practical large-scale ubiquitous computing system, we argue that lightweight and off-the-shelf mobile devices should be used by the user to interact with context-aware systems. Thus we employ the local infrastructure together with the home Personal Server system to do most of the work to support user's interaction. We have also proposed a solution for protecting the user's privacy based on an intermediary Anonymizer Proxy Server using pseudonym mechanism. We will further implement and evaluate our system to observe what and how much user will appreciate from the decreased interactions by using these intermediary proxy-based services in the infrastructure.

References

1. Mark Weiser: The Computer for the Twenty-First Century, Scientific American, pp. 94-10, September 1991
2. A. Fox, B. Johanson, P. Hanrahan, and T. Winograd: Integrating Information Appliances into an Interactive Workspace, IEEE Computer Graphics and Applications, May/June, 2000, pp. 54-65.
3. Patrik Werle, Fredrik Kilander, Martin Jonsson, Peter Lönnqvist, and Carl Gustaf Jansson: A ubiquitous service environment with active documents for teamwork support, UbiComp2001, LNCS, pp. 139–155.

4. Manuel Román et al: Gaia: A Middleware Infrastructure to Enable Active Spaces, IEEE Pervasive Computing, pp. 74-83, Oct-Dec 2002.
5. http://psi.verkstad.net/ACAS/
6. Ginger Myles et al: Preserving Privacy in Environments with Location-Based Applications, IEEE Pervasive Computing, January-March 2003 (Vol. 2, No. 1) pp. 56-64.
7. W3C: A P3P Preference Exchange Language 1.0 (Appel 1.0), working draft, WorldWide Web Consortium, Apr. 2002, www.w3.org/TR/P3P-preferences
8. Xiaodong Jiang , Jason I. Hong , James A. Landay: Approximate Information Flows: Socially-Based Modeling of Privacy in Ubiquitous Computing, Proceedings of the 4th international conference on Ubiquitous Computing, p.176-193, 2002.
9. J. Cuellar, Joel Morris, and D. Mulligan: Geopriv requirements. Internet draft, Internet Engineering Task Force, March 2003. Work in progress.
10. A. Harter et al: The Anatomy of a Context-Aware Application, Proc. 5th Ann. Int'l Conf. Mobile Computing and Networking (Mobicom 99), ACM Press, New York, 1999, pp. 59-68.
11. J. Rosenberg, Henning Schulzrinne, G. Camarillo, A. R. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler: *SIP: session initiation protocol*. RFC 3261, Internet Engineering Task Force, June 2002.
12. Jonathan Rosenberg: A presence event package for the session initiation protocol (SIP), Internet draft, Internet Engineering Task Force, January 2003. Work in progress.
13. Internet Mail Consortium: vCard Specification, http://www.imc.org/pdi/
14. Bluetooth.org: Bluetooth Specification Volume 1, Core 1.1, February 2001
15. J. Lennox and Henning Schulzrinne: Call processing language framework and requirements. RFC 2824, Internet Engineering Task Force, May 2000.
16. http://www.vovida.org/
17. http://www.xten.com
18. K. El-Khatib, N. Hadibi, and G.v Bochmann: Support for Personal and Service Mobility in Ubiquitous Computing Environments, EuroPar 2003.
19. Stefan Berger, Henning Schulzrinne, Stylianos Sidiroglou, Xiaotao Wu: Ubiquitous computing using SIP, 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'2003)