IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555
Vol. 2, No.3, June 2012

Enhanced Privion Privacy Protection Context-Aware Architecture

Mr.S.M.Krishna Ganesh
Department of Computer Science and Engineering.
St.Joseph College of Engineering & Technology
India.

Dr.S.Venkatesan Jayakumar
Department of General Engineering
St.Joseph College of Engineering & Technology
India.

Abstract— **Privacy is the ability of an individual to keep related information, out of public view, traditionally, privacy enhancing techniques have been used as a tool for hiding and control disclosure. It is necessary to think about how technology can be used to create visibility and awareness of information. We proposed an architecture in which a user can configure his personal preferences for privacy preservation functionalities and services, once he enters an UbiComp environment. The design is based on an analysis of various approaches to privacy protection for UbiComp settings. A composite entity called "privon" is introduced to encapsulate related information based on their privacy-sensitivity levels. A privon allow users to understand how their personal information may be used by others.**

Keywords- privon, privacy, Ubiquitous computing, context-information, USE tool.

I. INTRODUCTION

World Wide Web has significantly changed our world. Several research issues and opportunities have emerged the technologies and evolution of electronic gadgets as laptops. The main ubiquitous computing challenges [10] are Location and context awareness. The addressing of these issues is a great challenge bringing out Mark Weiser's vision [11]. Our interest in this paper focuses in context aware applications. The services provided by the Location and context aware systems are emergency services, security services, information and navigation services, billing systems, resource tracking management; and entertainment and proximity services.In this paper Privon, a data entity is integrated in to the architecture that provides personalization preventing the user contexted information until the user decides it. Policies have been defined as the 'rules that govern the choices in behavior of a system [12].The ubicom environment keeps track of user gather more information to perform their tasks better. There is a great threat to the user's privacy when more application interacts with the user [13]. Handling personal privacy is a key challenge in deploying ubiquitous computing services. Designing a framework for ubicom environment should seamlessly adapt to the highly dynamic environment. The scenario in next part enlightens the capability of the framework to adapt itself with the environment: Ubiquitous Computing can be thought of as the idea of invisible computers everywhere. Specifically, it is the idea that computers are embedded in the environment, with literally dozens or hundreds of computers available to each person, and each computer performing its tasks without requiring human awareness or a large amount of human intervention.

II. MOTIVATION AND BACKGROUND

Deploying a service on a significant scale for handling personal privacy is a key challenge in ubiquitous computing environment. in an architecture for privacy sensitive ubiquitous computing, the author focus privacy based on anonymity rather than considering everyday scenario.The key challenge to provide a framework capable of adapting everyday life scenario and provide privacy to each and every user in an highly dynamic ubiquitous environment, so the scenario illustrates these points. Sam assigned to several projects and has different access rights depending on the project. He can play any role in the software organization such as project manager, project leader, project developer, etc, having extensive preferences and email notification options. He submits the activities in the form of timesheet. On playing the role as project manager, Sam entail allocating new project, compile task details, department details, activity entry and timesheet entry. In New project he assigns a time frame for completing a project within the stipulated time where each member in the team is selected and one member will act as a Team Leader for the given project where he reports the project status to the project manager. In Task details each members will be assigned a module and assign a start date and end date for the given module where the timesheet submitted by the members of the project be verified by him to check the status of the project. In Department Details he can allocates the various departments in the organization and in case of expansion can add new departments. In Activity entry he monitor each member during no work hours like Meeting, Personal, Tea, Lunch, Sick leave etc., In Timesheet entry, check the details submitted by the members of the project and if the details are correct he will accept it or he will reject it.Project Leader will check for the task assigned for him and he/she will have a deadline for completing the project. He selects the task and completes it and after completing it, he will fill the Timesheet for the given module and he will enter

the details for each week, he will enter the details like the No of hours he worked, non working hours and what are the activity he involved during the working hours etc.,Once he fills the Timesheet he will send it to Project manager for acceptance. Also the project leader can access the details like the timesheet submitted by the project members. Project developer will check for the module assigned for him and he/she will have a deadline for completing the project. He selects the module and completes it and after completing it, he will fill the Timesheet for the given module and he will enter the details for each week, he will enter the details like the No of hours he worked, non working hours and what are the activity he involved during the working hours etc., Once he fills the Timesheet he will send it to Project manager for acceptance. Once if the timesheet submitted by the member is denied by the project leader, the timesheet status is set to open and the member have to resubmit the timesheet in order to proceed with the next timesheet.

A. Issues in Ubiquitous Computing

This is an overview of the current aspects of computing that need to change in order to help make ubiquitous computing a reality. We know that ubiquitous computing will involve many small devices, with many of them in constant communication with each other,so what kind of infrastructure is needed to make this possible? How can this kind of infrastructure are built on what already exists today? Here, we take a look at a few of the hardware and usability limitations that currently prevent ubiquitous computing from becoming a reality.

B. The Expense of Computing Devices

One of the fundamental ideas to ubiquitous computing is "one person, many computers". In order to fulfill the idea of many computers, each computer needs to be low in cost. Right now, a standard desktop computer costs somewhere between one and two thousand dollars, and hand-held computing devices (such as Apple Newtons or Palm Pilots) usually cost somewhere between two and five hundred dollars. This cost is obviously too high to allow for much beyond one computer per person.The kinds of computers that will be involved in ubiquitous computing, however, will not need to be as general purpose computers, and will not have the speed and storage requirements that general purpose computers have. As a result, they will cost less, especially since the cost of processors and hard drive are continously dropping in price, as part of the now famous "Moore's Law", which states that the amount of transistors that can fit on a piece of silicon doubles every eighteen months.There are still components that have not dropped significantly in price as time has gone by. Specifically, standard computer monitors and LCD screens have not dropped enough in price to allow for "throwaway" computer devices. In order for ubiquitous computing devices to have a small enough cost, they will need to be produced in the same volume of scale that, for example, electric motors. Ubiquitous computing will be feasible only when the cost of computing devices has dropped so low that individual computers with display capabilities, can be literally thrown away without the consideration of cost. As a comparison, hand-held calculators were once expensive devices, but are now so inexpensive that they are commonly given away as promotional products.

C. Lack of sufficient bandwidth

One of the requirements needed for ubiquitous computing to become commonplace is enough network bandwidth to allow the hundreds of devices needed to be able to communicate with each other. We are still in the early stages of networking, but as time goes by, more and more of the World's voiced based networks is being replaced by fiber optic and wireless networks. As an example of the network requirements that the World will face, a recent study by MCI estimated that the worldwide amount of data traffic will overtake the amount of existing voice traffic within three years, with the percentage only increasing in the future. One of the first things that new computer users need to learn is the concepts of files, and hierarchy of directories that contain them. Rather than concentrating on the information that is available, the user is forced to think in terms of how the information is stored. As a result, the information is stored in a non- intuitive manner, thus causing the user to lose track of the location of files, overwrite newer versions of files with older versions, accidentally delete directories containing important files etc. Almost every computer user has their own horry story of destroyed files, and in almost every case, it was due to a simple slip that should have been recoverable, but resulted in the loss of data due to the nature of file systems. One of the key ideas of ubiquitous computing is that computers should become invisible, and that they should provide information to the user in the user's own terms. For example, when a user wishes to work on a report from the previous day, they should not be forced to remember something such as "~/report.txt", or "C:\report.txt", since that forces the user to think in the computer's terms. It would be much simpler for the user, if they were able to say "open the report from yesterday". Such an ability, of course, requires high quality voice recognition, which is just now becoming feasible on common computers. But, whatever the access scheme may be, the important concept is that the user must be able to access data without needing to know specific file names or file locations, both of which are currently required. The user should be able to access data strictly by content, without worrying about the file format, file location, or even on which machine a file is stored. Right now, almost every software product in existence needs to be formally installed. That is, the computer needs to go through a strict process of reconfiguration in order to run the new software properly. Shared libraries need to be installed in the correct paths, global settings need to be changed (sometimes forcing the reboot of the computer), default user preferences need to be set up for the first time, and so on. In almost all cases, the installation process needs to be manually attended by a person, in order to make sure that the process goes smoothly, and in some cases, to make sure that the software being installed isn't some kind of security attack,

such as a virus. The only exception to this is embedded firmware programs, which are burned in as part of the hardware. In order for ubiquitous computing to happen, the concept of installation will need to disappear. Rather than the person going to the software, the software will need to go to the person, and as a result, programs will need to be able to flow freely from computer to computer, without requiring that each computer make fundamental changes to its own configuration in order to run the new program. Computers will need to properly partition off each program, so that a new program will never be able to grab unlimited amounts of resources, or access files that aren't properly related to the program's function. Also, once a program moves from one computer to another, all traces of its existance should be removed from the previou computer, such as user preferences. The usage of software will need to become "stateless", where each program is always ready to perform its given function at any time, without needing to be configured first. A close analogy to this is the common ASCII text file. Text files can be trivially moved from one computer to another, no matter which operating system is running: Unix, Windows, Mac,OS/2, etc. There is never any worry of the need to "install" or "deinstall" a file, and there is never any worry of the file being able to corrupt the system. For ubiquitous computing to succeed, programs will need to be the same way: An easily transferable entity, written in a universal format.The closest possible candidate that we have now for this "universal format" for programs is Java. There are two very strong points in its favor. First, the format used for storing code is processor independent, thus making it well suited for transferring from one machine to another. The issue of what kind of processor a machine uses (Intel, PowerPC, SPARC, Zilog, StrongArm,etc) becomes irrelevant. Second, security is built into the format, in that it is theoritically impossible for a program to make system calls that shouldn't be allowed. This makes it very well suited for transferring, since no Java program will be able to violate the security of the machine it is hosted on. The largest downside for Java is that its programming interface is still in flux,thus making it impossible to write programs with taking versioning into account. It won't be possible to use Java for ubiquitous computing until one of two things happen: Either the programming interface is frozen in place (which, given the history of computers, is very unlikely), or some kind of system is established where a programming interface can be carried along by the program that uses it. No doubt, this will be an important part of ubiquitous computing research as it grows in importance. The paper is structured as follows. First, we review related work in Unicom privacy, followed by a brief introduction to Altman's theory. We then continue by identifying the similarities between the framework and Unicom, extending the model when necessary, and apply the result to typical Unicom use cases. Finally, we discuss the success of the model and point out future directions.

III. RELATED WORK

Privacy assistant, a personal trusted device used in a privacy-awareness system paw suffer limitation on sensor configuration in dynamic environment. But Langheinrich [5] provides some degree of transparency on the collection and usage of sensitive personal information.An architecture developed by Hong and Landy [6] for privacysensitive Unicomp, Confab that captures information on user device. This architecture implements a social component of privacy protection to hide their real privacy preferences. The architecture does not hold Unicom environment as context information of the user cannot be acquired by external sensors. Ubiquitous computing environment consist of heterogeneous devices where realizing the repository becomes complex. Weust [7] designed a middleware focusing on serviceorientation. The system should focus more on usability and inter-operability than innate characteristics. The approach in [8] relies on the assumption that data is only collected locally. It's an unrealistic assumption in Unicomp environment.The USE tool mainly used for specification and validation of RBAC policies. The USE system provide authorization engine for validation purpose of RBAC policies. K.Sohr [9] employs various ways in the context of RBAC policies.

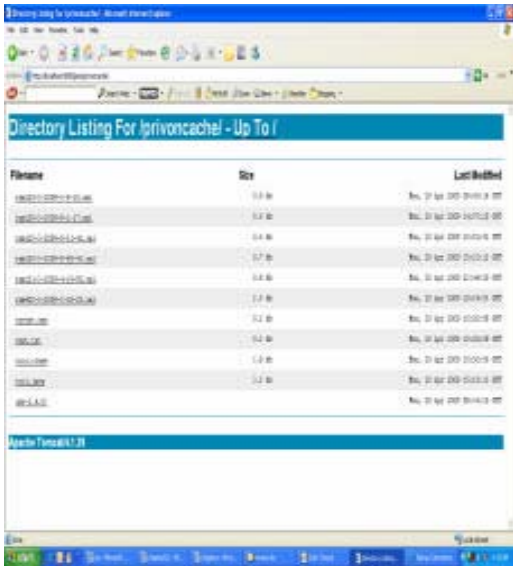
IV. PROPOSED PRIVACY ENHANCED SYSTEM ARCHITECTURE

A general architecture for ubicom environment is developed. This architecture for context aware system to infuse privacy into it without any special assumptions on hardwares where applications run. A middleware developed to provide a user the ability to configure his personal references for privacy preservation and is a service-oriented enduring limited device capability that handles the dynamic nature of ubicom systems. A composite entity called "privon" that encapsulates related information based on their privacy sensitivity levels in the middleware. A privon is envisioned to allow users to understand how their personal information may be used by others, and to understand how they and their information participate in the system.

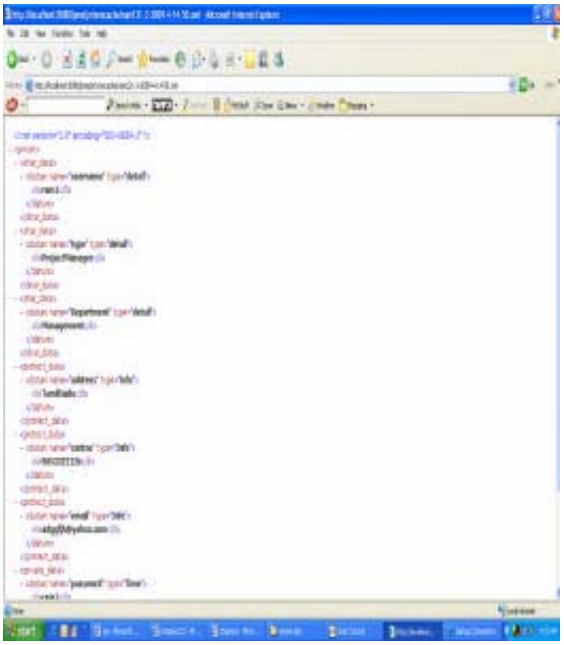
V. IMPLEMENTATION

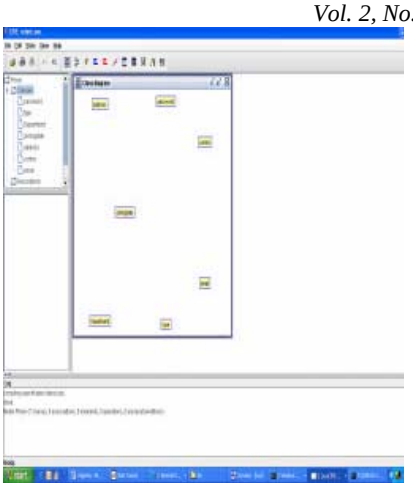
This architecture works well with the scenario. Each user has given preference to configure his own information. The user i.e. Sam has keyed his information in the server. The context server allows providing data in several levels of granularity and anonymity. Building on this, the access control component ensures that personal data can only be accessed by authorized parties or services. Once data is disseminated, the virtual environment component provides appropriate information flow control. The identity management component ensures that users may be pseudonymous against the Ubi- Comp infrastructure.The traceability facilities are interwoven with the privacy protection mechanisms;they are part of the access control,identity management and Transparency management components. The access control mechanisms are designed to support fine-grained specification and delegation of access to personal data. Users are enabled to grant and to delegate access to groups and hierarchies of parties or services, enabling a distribution of responsibilities. The identity management enables privons,that may be traced back to a core

identity, or to a group membership. In order to be able to balance between privacy and traceability, users must be able to express their preferences about which data to disclose in which circumstances, in a fine-grained manner.For this purpose, fine- grained resolutionof context data, of identity information,and of access rights are provided. The USE system is employed to check the generated privons that help the policy architect design and express higher level organizational rules.



In the above figure shows the privon cache stores the privons generated during each session. The Following figure shows the Data elements displayed in XML format.





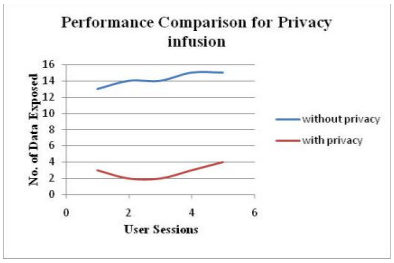
The above figure shows the generated privon checked by the USE system. The figure below shows the Smart Work place scenario implemented using J2METM for mobile devices.

Session ID	Data	Weight
010CB162122DCA08918D8DF2BC5FE537	Password	0.14285715
010CB162122DCA08918D8DF2BC5FE537	Type	0.14285715
010CB162122DCA08918D8DF2BC5FE537	Contno	0.14285715
010CB162122DCA08918D8DF2BC5FE537	Address	0.2857143
010CB162122DCA08918D8DF2BC5FE537	Joiningdate	0.0
010CB162122DCA08918D8DF2BC5FE537	UserName	0.14285715
010CB162122DCA08918D8DF2BC5FE537	Email	0.14285715
010CB162122DCA08918D8DF2BC5FE537	Department	0.14285715
13A66FB838605049451C4FEC806B750B	Password	0.16666667
13A66FB838605049451C4FEC806B750B	Type	0.0
13A66FB838605049451C4FEC806B750B	Contno	0.33333334
13A66FB838605049451C4FEC806B750B	Address	0.33333334
13A66FB838605049451C4FEC806B750B	Joining Date	0.5
13A66FB838605049451C4FEC806B750B	UserName	0.16666667
13A66FB838605049451C4FEC806B750B	Email	0.0

TABLE PERFORMANCE EVALUATION FOR PRIVON WEIGHTAGE

Table shows a portion of the table in the privon cache. Table captures the advantage of using privons in a ubiquitous environment. We have considered a set of events in which a user requests for information. In the absence of privons, this information is transparent to users in the environment on deploying privons, the nature of data changes, depending on the privacy level set by the user. To illustrate data classification in our scheme, let us consider an entry for a

data element in Table. Let us consider $w1$ = 0.14285715 of session ID 010CB162122DCA08918D8DF2BC5FE537. The weight implies that $u1$, the data element corresponding to $w1$, is classified as protected data. Let us assume that the user decreases the privacy level of the session by ± = 0.1 Then, Dec0.1 (w1) = 0.4 changing $u1$into transparent data. On the other hand, if the privacy level is increased by ± =0.1, Inc 0.1 ($w1$) = 0:3, making $u1$private.



Above graph reveals a frequent definition of information privacy is "*the claim of individuals, groups or institutions to determine for themselves when, how, and to what extent information about them is communicated to others*".

CONCLUSION

This paper provides a work on privacy in ubiquitous computing environment. In this paper a data entity called privon is used to provide the abstract nature of personal privacy of a user. We have implemented the Smart Work place scenario using Java TM technology for Enterprise Applications, and J2METM for mobile devices. The session history lists the time, date, requestor, and privacy level of any prior request. A slider is provided, by which the user can set the privacy level for the entire session. The user can also view the data currently designated as transparent, protected, and private and make modifications if necessary. The USE system is employed to check the generated privons that help the policy architect design and express higher level organizational rules. This proposed work is a part of the work on Privacy Enhanced Context-aware Information in ubiquitous computing environments. As the Future work of this project the generated privons can be checked for authorization constraints help the policy architect design and express higher level organizational rules. The future work will be concentrated both non-temporal and history-based authorization constraints with the use of Object Constraint Language (OCL) and first-order linear temporal logic (LTL). In summary, we strongly believe that both formal and practical approaches for policy specification and verification are needed in the design phase to rule out any inconsistencies and undesirable properties of RBAC policies at the early stage.

REFERENCES

[1]. Karsten Sohr,Michae rouineaud,Gail Joon Ahn,senior member,IEEE, and Martin Gogolla, Analyzing and managing Role-Based access control policies, IEEE transactions on knowledge and data engineering,Vol 20,No 7,July 2008.

[2].Stefan G. Weber, Andreas Heinemann, Max M¨uhlh¨auser, Towards an Architecture for Balancing Privacy and Traceability in Ubiquitous Computing Environments The Third International Conference on Availability, Reliability and Security,2008.

[3].Gautham Pallapa, Mohan Kumar and Sajal K. Das, Privacy Infusion in Ubiquitous Computing, 2008.

[4].Gautham Pallapa, Nirmalya Roy, and Sajal Das, Precision: Privacy Enhanced Context-Aware Information Fusion in Ubiquitous Healthcare, First International Workshop on Software Engineering for Pervasive Computing Applications, Systems and Environments (SEPCASE'07),2007.

[5].M.Langheinrich.A Privacy Awareness System for Ubiquitous Computing Environments. In Proceedings of the fourth International Conference on Ubiquitous Computing (Ubi-Comp 2002), pages 237–245.Springer-Verlag, Sept. 2002.

[6]. J. I. Hong and J. A. Landay. An Architecture for Privacy-Sensitive Ubiquitous Computing. In Proceedings of The Second International Conference on Mobile Systems, Applications, and Services (MobiSys 2004), 2004.

[7]. Wuest, B., Drogehorn, O., and David, K., Framework for Middle ware in ubiquitous computing systems, IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2005, vol.4, Page(s):2262 – 2267, 2005.

[8].C. Troncoso, G. Danezis, E. Kosta,and B. Preneel. Pri-PAYD: Privacy Friendly Pay-As-You-Drive Insurance. In Workshop on Privacy in the Electronic Society 2007 (WPES '07), 2007.

[9].K. Sohr, G.-J. Ahn, and L. Migge, "Articulating and Enforcing Authorisation Policies with UML and OCL," Proc. ACM ICSE Workshop Software Eng. for Secure Systems (SESS '05), May 2005.

[10].Mark Weiser, The future of ubiquitous computing on campus, Communications of the ACM, vol. 41, No. 1, Page(s):41–42, 1998

[11].R. Agrawal, A. Evfimievski, R. Srikant, Information sharing across private databases, In Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, Page(s):86-97, ACM Press, 2003

[12].K. Sohr, G.-J. Ahn, and L. Migge, "Articulating and Enforcing Authorisation Policies with UML and OCL," Proc. ACM ICSE Workshop Software Eng. for Secure Systems (SESS '05), May 2005

[13].A. I. Gonz´alez-Tablas, L. M. Salas,B. Ramos, and A. Ribagorda. Providing Personalization and Automation to Spatial-Temporal Stamping Services. In DEXA Workshops, pages 219–225,2005.