ISSN:0975-9646

Secured Agile Architecture for Context Aware Pervasive Computing

Alti Siva Prakasa Rao[1] and Prof. M.S.Prasad Babu[2]

[1]*Department of Computer Science & Systems Engg., Andhra University, Andhra Polytechnic, Kakinada, East Godavari Dist. A.P., 533 002*

[2.] *Department of Computer Science & Systems Engg. Andhra University, VISAKHAPATNAM*

***Abstract*- Context refers to information such as location, time, user preferences, user emotions, environment status, computational devices and network parameters etc. Context-aware pervasive computing utilizes the information about the physical world collected form various sensors and changes the behavior automatically considering current context, history and other user preferences. The context viewed or used by pervasive systems may change mobility or characteristics of that particular device. Access of certain information may be restricted to a particular security level to protect information from tampering. Security contexts enhance the security in context-aware pervasive systems which are nothing but collection of user's information which is applicable to available security policies. Traditional security models aim to provide integrity, confidentiality and availability of information. In Pervasive computing environments high level personal privacy is very much required. In this paper a new architecture is proposed for context-aware pervasive computing using agile security concepts. It explores how best agile software agents can adapt information and modifies the context as per security policies.**

Keywords: Software architecture, Secure pervasive computing, Ubiquitous computing, agile, software framework, context awareness, re-factoring.

1. INTRODUCTION:

Context–aware pervasive computing is becoming much prominent field in the new era of communications. It is important to discriminate between real world situation, the data captured to represent situation, the abstract representation of the situation and mostly behavior of the application according to the available information [1]. The context depends on situation, sensor data, history etc. Moreover the context-aware information used for determining the context, should be error free and time based or location based or both time and location based. When implementing the context awareness idea, it should use the sensor data and predict the current situation or even forecast a situation by using proper probabilistic models and data mining techniques. Also while capturing sensor data it is needed to identify the useful contexts for the computing environment. As a variety of context-aware environments begin to grow, pervasive applications shall have to interact with environments that they have never encountered before. Extension to Strathclyde Context Infrastructure (SCI) [2] gives context-aware applications, the potential to adapt unfamiliar environments transparently. It is actually a vision of a context discovery based on automated semantic reasoning about context information and services.

The Context Modeling Language tool (CML) [3] will assist designers with the task of exploring and specifying the context requirements of a context application. The model will capture user activities as temporal fact, association between users and communication channels and devices, locations of users and devices. An intangible framework and software infrastructure that together address known software engineering challenges and enable further practical exploration of social and usability issues by facilitating the prototyping and fine tuning of context aware applications [4]. In particular, pervasive computing demands applications that are capable of operating in highly dynamic environments and of placing minimal demands on user attention.

2. AGILE SOFTWARE ENGINEERING ENVIRONMENT:

Creating solutions of any size based on pervasive computing requires an understanding of the entire software life cycle. It is no longer acceptable to just hope the final system comes together. Instead, designing for deployment, performance and security is necessary at the outset. This session looks at the new Software Development Framework Agile, a context-based software development process for building context-aware pervasive computing applications. It will concentrate on practical and concrete techniques that can help to build better-performing, secure, and deployable solutions.

In the mid of 1990s the modern definition of agile software development was evolved as part of a reaction against "heavyweight" methods, as typified by a heavily regulated, regimented, micro-managed use of the waterfall development model [19]. The process is originated from the use of the waterfall model and was seen as bureaucratic, slow, demeaning, and contradicted the ways that software engineers actually perform effective work.

Initially, agile methods were called "lightweight methods". In 2001, prominent members of the software community met at Snowbird and adopted the name "agile methods". Later, some of these people formed 'The Agile Alliance', a non-profit organization that promotes agile development. Through agility it makes fundamental assumption that continual change should be expected and that is impossible to isolate a solution (context) from these changes. It recognizes the context requirements that can be

difficult to articulate at the outset and that they will often undergo significant modifications as the possibilities become clearer to users. Agile process is iterative process which will show the status of quality progressively. Agile methodologies develop practices that are most useful and primarily applied during the design and development phases of a project life cycle.

The agile approach like specification–driven development combines features of test-driven development and plan-based approach of design by contract, will lead to development of context-awareness in pervasive computing [8]. Both tests and contracts are different types of specifications (contexts) and both are useful and complementary for building high quality solutions. For designing tests or contracts are similar to UML's class and collaboration diagrams. Extension to this agent based approach will combine high level functions with static typing to separate the routine from functions.

Particularly agile software engineering environment will support Just-In-Time (JIT) management, network-centric architecture and collaborative features (distributed repository), which are not easy to implement in conventional software engineering environments [10]. In agile methodology, it is important to understand that it is not possible to understand all the requirements at the beginning of project, then analyze these requirements and finally to develop the architectures for the whole system [18].

3. RELATED WORK AND MOTIVATION:

The oxygen project [6] at MIT envisions a future where the ubiquitous computing devices are freely available and easily accessible as oxygen. The goal of project AURA [7] at Human Computer Interaction Institute of Carnegie Mellon University is to provide users with invisible ambience of persisting computing and information services regardless of location. Planet Blue [11] project at IBM and cooltown Project [12] at HP. Jini is a dynamic open-architecture networking technology enabling spontaneous networking of services. Jini enables ad-hoc network information through mechanisms such as service discovery/join and distributed events. 'One World Architecture' requires that all resources should be explicitly bound and released while ISAM supports dynamic adjusts to new state based on system application negotiating [18].

None of the architectures addressed all the security requirements like authentication, authorization and confidentiality. Most of the literature focuses on the intrinsic limitations of environments and connected devices. Also addressing the security requirements for all kinds of environments (heterogeneous) and context would be too expensive or not possible on tiny computing devices [21]. An autonomous software agent [like Agile Process Agent] will transfer its execution dynamically, save network bandwidth and increase the reliability and efficiency of the execution. These agents will be capable of performing the activities like maintainability, interoperability, scalability, reusability, extensibility, portability and adaptability [22].

It is very interest to observe that all previous work on combining security and context aware pervasive computing applications with focus on specific applications only. They never considered user's preferences. The present Agile Process Agent will consider the user's preferences also in the form of predefined contexts. Also security decisions follow an old fashioned rule-based formalism which does not consider systems and network dynamics. No architecture addressed the reusable architectures or mechanisms for context acquisition and storage as mentioned in the proposed architecture [17]. Using data-centric data model, such as XML schema, expresses schemas as data rather than code.

4. PROPOSED ARCHITECTURAL MODEL FOR CONTEXT-AWARE PERVASIVE COMPUTING:

Agile processes encourage and embrace requirements change. The proposed architecture consists of three basic components namely context information model, network centric database server and Agile Process Agent.

4.1 Context Information Model:

This model will emphasize the modules like sensor data collection, categorization of collected data into distinguished contexts, modeling the context data using the contexts. On selecting the related context model and by using the data mining tools delivers the services if model matches with the concerned context model. The better aggregation of context types will help application designers to uncover the most likely parts of context that will be useful in their applications [28]. Randomized Cache prediction algorithms interpreted in this model improve the cache hit rate in the case of disconnected operation for pervasive devices like handheld mobile devices. The algorithm includes phases like data pre processing, clustering the data according to association degree and selection of data required for applications. Provision for cache replacement algorithm enhances the effectiveness of the model. The context modeling module incorporates the components like cache algorithm, clustering procedure and data mining tools.

4.1.1 .Cache algorithm:

The goal of cache maintenance algorithm is to minimize the cache misses which are truly expensive part of process. The sequence of memory references is not under control of the algorithm and it depends on the application that is running. Cache replacement algorithms like LIFO, FIFO, and LRU are used to predict the cache.

4.1.2 Clustering for context model:

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. A cluster is a collection of data objects that are similar to one another within the same cluster and dissimilar to the objects in other clusters. For context modeling, k-medoids algorithm may be used for identifying the similar contexts.

4.1.3 Data retrieval:

Selection of correct context may be obtained by using the techniques like Apriori Algorithm for finding frequent item sets using candidate generation. Apriori is an influential algorithm for mining frequent item sets for Boolean association rules.

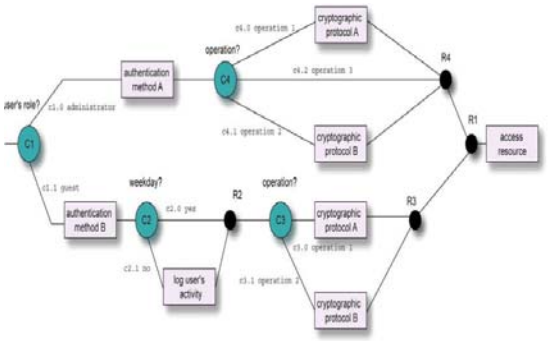
4.2 Network centric Database Server:

The graphical notation of CML (Context Modeling

Language) is well suited for use when defining the context information used by context aware applications, and its relational analogue is a natural choice of context storage and management [4].Data can be maintained as relational database. The metadata will contain the indexed quick reference to the various parts of Database server. The conceptual design of data base involves forming effective relations between temporal data and active dataset. The need of a distributed, intelligent and adaptable context aware system is felt that facilitates resource discovery and adaptation, as indicated by the user demand. A history associated with each user predicts the user's preferences for different interactions. A distributed model of a context aware system that adapts services on the basis of the inferred contextual information coupled with the history of the user is thus envisaged. Contextual graphs elaborate the security concerns well.

4.2.1 Contextual graph:

Contextual graph is an acyclic graph with a unique input, a unique output and a serial parallel combination of nodes connected by oriented arcs. By using contextual graphs it is easy to identify the roles and authentication of users identified. Example shown below.



4.3 Agile Process Agent:

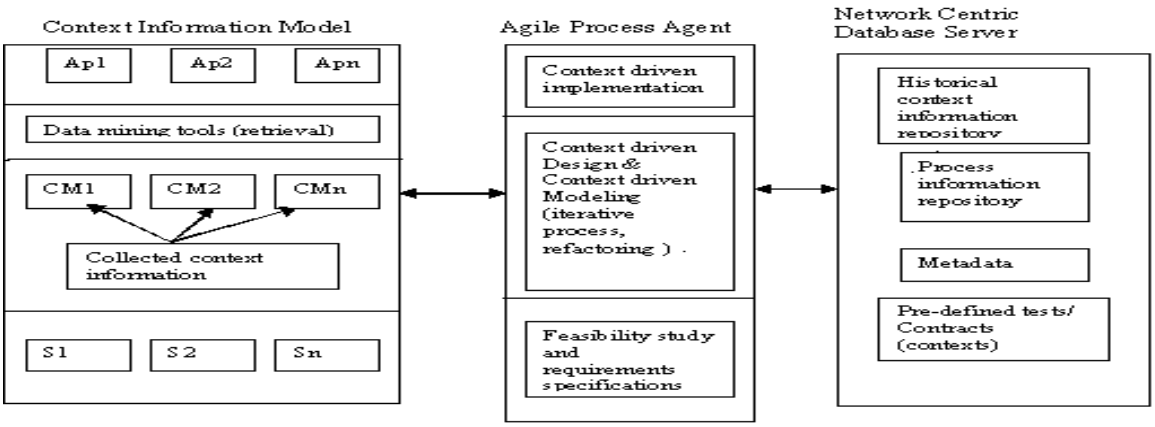
The Agile Agent, which provides the core agile agent development process, implements an Integrated Development Process (IDP) that consists of four key phases: Design, Test-Driven Implementation, Release and Review, and Re-factoring and Enhancement that are applied iteratively until a finished state is reached. The Agile agent supports a bottom-up approach that increases flexibility and enables the development team to focus on the rapid delivery of working model, and to respond quickly to changes in requirements. Re-factoring and Enhancement is not strictly a phase; instead it represents a continuous process that involves applying improvements and enhancements to the finished model. Typically, re-factoring entails applying changes to the internal structure of the application software to make it more efficient and easier to comprehend, without altering the external behavior. This process may necessitate: removal of duplicated code; the simplification of complex logic; and clarifying of ambiguous model. This continuous analysis of model helps enhance the design and implementation of the application (re-factoring in small steps helps prevent the introduction of defects). Enhancements include new or improved functionality that may further improve the application. This process, along with the review, highlights outstanding issues and motivates any required enhancements to the process.

Agile Process Agent introduces the agility (adoptability) in acquiring context information and responding to the applications in real time. The agile process agent works with iterative and refactoring features. Agile Process agent reflects the all phases of agile framework. It exhibits the requirement analysis and feasibility study processes in analyzing sensor data collection. The central block of agile agent exhibits context driven design & modeling with the help of refactoring feature. It is similar to the Specification Driven development [8]. After collecting the context information from a set of sensor data aggregated on specific preferences in the context model. If any context seems to be new that context information will be allotted with new name and updated the list of contexts in the database. Context models are prepared focusing on some preferred specifications only. Clustering algorithms [5] may be used for context modeling. This process identifies outliers in context information and makes the model stable. Also it is possible to use context information for predicting model which is not available in the database. Refactoring the existing model leads to improve the efficiency. Since refactoring leads to redesign from the bottom up, often to eliminate model deficiencies, modules may be assigned new functionality which may not work well with security constraints [23].

Once the modeling done, with the help of sophisticated data mining tools like apriori algorithm, the context model responds to user requests quickly. While responding to user request and context model selection the agile agent works collaboratively with database which is residing in a network centric database server. Once connected to the user the agile process agent keeps on iterating the modeling and design modules to maintain the quality of service, flexibility and adaptation. Kick-off services [13] provide the support and guidance to quickly overcome obstacles and gain proficiency in their agile practices. This model provides a feature like default model assignment, since even after the construction of excellent models from context information by the Agile Process Agent they may not match the requirements of the application request by the user. In those conditions the generic models which were kept in the database will be assigned to that application. But during the period of application usage, the agent tries to improve the quality of model through refactoring feature.

5. AGILE SECURITY:

Agility is a set of combined quality properties, and sensitive to varying resources and sensitive to changes in contexts. Beznosov presented a conceptual analysis of the suitability of 'Extreme Programming' for building secure systems and also introduced the notion of "good enough security" without defining it a priori [20]. Iterations and frequent iterations common to agile processes significantly increase the cost of involving the third party at all iterations, while the security assurance efforts proceed, development should continue [23]. Now most of the research is towards securing context-aware applications. Contextual information will enhance control of the model



Ap1, Ap2…Apn: Applications,. S1, S2… Sn: Sensors CM1, CM2, CMn: Context Models

Fig: 1 Proposed architectural model

which will further focus on the specifications of the applications [17]. While taking security decisions the methods like rule based systems may be considered. But rule based systems can not solely provide security for the applications which are in dynamic nature. Most of the times model may visit unknown contexts, at that time the agile agent will allot new name to the new context. Then refactor the new context using the predefined context information and metadata [15]. Agile agent maintains security in three levels. Context capture, categorization, modeling and allocating to proper applications are the main

tasks in the present architecture. The next important feature of Agile Process Agent is using the risk assessment techniques to make informed choices as to which requirements are to be addressed and computational / probable security type is deserved. The proposed system utilizes both rule based and probabilistic based decision support systems for building valid security policies. The following shows proposed architectural model.

5.1 REQUIREMENT ANALYSIS:

This process determines the important tasks required to be implemented and unwanted tasks to be stopped. Each requirement can be designed as a separate feature (description of the required task). Also unwanted features will be designed to secure the system from attackers. In this phase Feature Driven Development (FDD) allows the model to capture the details like potential security objects. Then the agent decides the unwanted actions on those secured objects (contexts). The important features will be updated into the network centric server for the later use of agent. Also in this module Agent will identify the malicious data captured from various sensors to reject the useless contexts by using security policies. While discovering the contexts the architecture will quickly react to the faults. This module also considers unambiguous or missing contextual information which will be rarely observed in other architectures. This process is the heart process of Agile Process Agent. Also it requires establishing security policies which hold a set of rules, principles and practices that determine how security is implemented. The potential risk contexts will be identified in this phase while using the probabilistic methods like statistical learning theory [25]. Through this method identifying the risk context and rejecting such contexts / features at the first stage of requirements itself will reduce the burden on the system. The security policy must be adaptable such that it can lead to create a new context for the changed information.

5.2 CONTEXT DRIVEN DESIGN AND CONTEXT DRIVEN MODELING:

This module models the standard contexts and maps the security constraints on each defined model. In this module authentication and authorization of the modeled contexts will be checked. Accessibility, time for accessing the information etc. will be mapped along with each context using Web Ontology Language (OWL-S). Ontology allows the model to map distinguished constraints on the model. A contextual graph allows a context –based representation of a given problem solving for operational processes by taking into account the working environment. Stephen S. Yau [16] illustrated that the context-based security policy can be used to manage distributed applications. Similarly Agile Process Agent tries iteratively for each context to adapt the security policy and model such contexts only which are authorized by the system. Predefined security policies also provided in the data base server as trusted contexts. If the exact match of requested context model is not available in database the agent will create a new model for the unknown model with new name and updates in the contexts list of database server. Agile agent observes the quick changes in sensor data for the running context and adopts the changes using JIT process.

5.3 CONTEXT DRIVEN DEVELOPMENT:

In this module Agile Process Agent will match the applications (services) to their concerned context models. Again for service delivery the agent should check the authentication by using security policy. Through authentication the agent can make sure that service is delivered to the intended recipient, which can in turn make sure that the services come from the known sources [21]. Security requirements are strictly contextually dependent than any other type of requirements.

5.4 SECURITY POLICIES:

Context aware security policies will contain the information regarding the authenticated user, location, time and other required contextual information. Whenever agent receives valid contextual information it will interact

with the database server to confirm whether it is a valid context or not using encrypted protocols. Depending on the match with security policy agent will take actions like approving the information and forwarding for contextual modeling or rejecting the information if it doesn't match with predefined security policy. Third type of action also the agent may take if the data is partially satisfying the policy constraints. Then Agent will apply the probabilistic approach to determine to which policy it may satisfy, later the data will be forwarded for context modeling module.
On the client side, the user should be sure that only services matching his preferences would be returned. From his point of view, trusting a service should therefore go beyond the simple authentication of the service provider and also encompass a complete certification process of the capabilities of the service. On the server side, the same problem is possible since the server does not know the users that can be trusted and can gain access to its service. They should therefore be accessible only to client they trust to access them according to a precise behavior guaranteed by some authority [26].The basic ideas to protect the privacy are enabling access and information to the authorized agents and protecting user's private space from un desirable interruptions provoked intentionally by system abuse [27].

5.5 A CASE STUDY: University Campus

The present case study illustrates the definition of security policy and updating the policies using adaptability feature of the proposed architecture. In this case, a part of locations in the University campus like Lecture theatre, Laboratory and Library are considered. Accessibility of resources is permitted in between 9am to 5pm. The resources available are i) LCD projector, ii) Printer and a iii) Fax machine. The designing of two security policies namely Policy A and Policy B for context security and updating are mentioned below.

Context security policy (Policy A1: Accepted):
Accept ((Locations->lecture theatre, Laboratory, library), (Time->9am to 5pm), (User ID->E1000 to 9999), (Network ID->N1000, N2000, N3000), (Service Request->LCD Projector, Printer, FAX))

Context security policy (Policy A) with function name 'Accept' contains five parameters to be satisfied namely Location, Time, User ID, Network ID and Service Request. Among all the parameters 'Network ID' is one of the parameters that must be satisfied for any service request. If an unauthorized user requests LCD Projector access at 12pm while staying at faculty quarters, the Agent will simply rejects the request. Since the first parameter Location doesn't match to satisfy the Policy A.

Context security policy (Policy A2: Partially Accepted):
Accept ((Locations->lecture theatre, Laboratory, library), (Time->9am to 5pm), (User ID->E1000 to 9999), (Network ID->N1000, N2000, N3000), (Service Request->LCD Projector, Printer, FAX))

In an instant if an employee requested the FAX machine to access from the library location at 5.02 pm. Even though the security policy A matches partially, the agent updates the security policy at database server since it understood that it may be an emergency case regarding that particular machine instead of rejecting the request. Policies both A and B will be kept in Data base.

Updated policy (Policy B:)
Update ((Locations->lecture theatre, Laboratories, library), (time->9am to 6pm), (user ID->E1000 to 9999), (network ID->N1000, N2000, N3000), (Service request->LCD Projector, Printer, FAX))

The Update Policy (Policy B) with function name 'Update' contains the same parameters as Policy B, but this function activated by the agile process agent when ever few of the five parameters are of policy A are met. This Policy Updates the context information and stores the new context in the database server for further use.

Table 1 will show the Agile Process Agent actions for different contextual information after applying the security policies A and B.

Table1 Agile Process Agent Actions for security policies

Context ID	Geographical location ID	Time	User ID	Network ID	Service requested	Policy match	Agile Process Agent action
C1000	Lecture theatre	9 am	E12345	N1000	LCD Projector access	yes	Forward data to next module
C1200	Railway station	10 pm	E9873	unknown	Printer	NO	Discard the context data
C1500	AI Laboratory	8 am	E3456	N2000	Fax machine	Yes (partial)	Update the policy

6. CONCLUSIONS AND FUTURE DIRECTION OF THE WORK:

In the present work the flexible architecture for the pervasive computing environment was thoroughly discussed. The security policies considered in the present architecture illustrates that agility as a key factor to demonstrate the better efficiency over the other existing architectures. Implementation, Database development, agile documentation and agile configuration management areas are yet to be explored for improving the response, as per agility myths. Improving the security concerns of this model by using Model-Driven Agile Development (MAD) [16] is another direction for the proposed architecture

7. REFERENCES:

[1] Albert Schmidt, Teco, Kriston Van Laerhoven., *How to Build smart applications*, IEEE Personal Communications, August 2001

[2] Graham Tomas, Sotioris Terzis and Paddy Nixon, *Towards Dynamic Context Discovery and Composition*, 1st UK-UbiNet Workshop. Imperial College. London, England. September 2003.

[3] K.Henricksen & J.Indulsks, *Developing Context aware Pervasive Computing Applications: Models and Approach*, Preprint submitted to Elsevier Science, 20th July 2005

[4] Karen Hendrickson, J. *IndulskaA.Software Engineering Framework for Context-aware Pervasive Computing*, Proceedings of the Third National Conference for Emerging Researchers in Ageing, 2004

[5] Paddy Nixon, Simon Dobson and Gerard Lacey, *Managing interactions in smart environments*, Springer Verlag, pp.243, December 1999.

[6] Project oxygen, http://oxygen.ics.mit.edu

[7] Project Aura, http://www-2.cs.cmu.edu/~aura

[8] Jonathan S. Ostroff, David Makalsky and Richard Paige. *Agile Specification Driven Development.* Fifth International Conference on Extreme Programming and Agile Processes in Software Engineering XP2004, June 6-10, 2004, Garmisch-Partenkirchen, Germany.

[9] Jonathan S. Ostroff, Richard F. Paige, David Makalsky and Phillip J. Brooke. *E-Tester: a Contract-Aware and Agent-Based Unit Testing* F*ramework for Eiffel*. Journal of Object Technology, Volume 4 Number 7, Sep-Oct 2005.

[10] Mikio Aoyama, *Web Based Agile Software Development*, IEEE software, November /December 1998.

[11] Planet Blue http://www.research.ibm.com/CompSci/planetblue

[12] Cooltown, http://cooltown/hp.com

[13] web site: www.agilelogic.com

[14] Paul Hodgetts, Denise Phillips and Oscar Chico, *eXtreme Development with the Java™ 2 Platform, Enterprise Edition (J2EE™)*, Javaone Sun's 2000 worldwide java developer conference.

[15] Mikko Siponen, Richard Baskerville and Tapio Kuivalainen, *Integrating Security into Agile Development Methods*, Proceedings of the 38th Hawaii International Conference on System Sciences – 2005, 0-7695-2268-8/05

[16] Stephen S.Yau, *Model Driven Agile Development (MAD) for Situation-aware Software in Ubiquitous Computing Environments*, Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05), 2005

[17] Ghitha Kouadri Mostefaoui, Contex-Aware Computing: A Guide for the Pervasive Computing Community, Proceedings of the IEEE/ACS International Conference On Pervasive Services (ICPS '04) – 2004

[18] Robert Grimm, one.world:Experinces with a Pervasive Computing Architecture, IEEE CS and IEEE ComSoc, July-september 2004, Pervasive Computing.

[19]http://en.wikipedia.org

[20] K.Beznosov,eXtreme Security Engineering. In Proceeding of First ACM Bizsec Workshop, Fairfax VA, USA, October 2003.

[21] Domenico Cortroneo, Almendo Grazizno and Stefano Russo, *Security Requirements in Service Oriented Architectures for Ubiquitous Computing*, 2nd workshop on Middleware of Pervasive and Ad-Hoc Computing, Toronto, Canada, 2004.

[22] Eila Niemala, Teemu Viskivuo, *Agile Middleware of Pervasive Computing Environments*, Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04), 2004.

[23] Konstantin Beznosov, Philippe Krunchten, *Towards Agile Security Assurance*, NSPW, NOVA

[24] Patrick Brezillon and Ghita Kouadri Mostefaoui, *Context-Base Security Policies: A New Modeling Approach*, Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications workshops (PERCOMW'04), 2004.

[25] Vladimir N. Vapnik, *An Overviiew of Statistical Learning Theory*, IEEE Transactions on Neural Networks, Vol 10, No 5, September 1999.

[26] Slim Trablsi, Laurent Gomez and Yves Roudier , Context Aware Security Policy for the Service Discovery, 21st International Conference on Advanced Information Networking and Applications Workshop (AINAW'07).

[27] Nuno Pratas, P.N. Anggraeni, S.A. Wardana, Neeli Reshmi Prasad and Ramjee Prasad, *Context Aware Trust and Privacy Application for Mobile Identification Sytem*, reviewed by IEEE Communications Society subject matter experts for publication on WCNC 2009 proceedings.

[28] Oleg Davidyuk, Istvan Selek and Jon Imanol Duran, *Algorithms for Composing Pervasive Applications*, International journal of Software Engineering and Applications, Vol 2, No.2, April 2008

[29] J.Madhusudanan, A.Selva Kumar and R Sudha, *Framework for Context Aware Applications*, 2010 Second International Conference on Computing, communication and networking technologies, published by IEEE Computer Society.

[30] Xiang Wang and Ian Davidson, *Discovering Contexts and Contextual Outliers Using Random Walks in Graphs*, 2009 Ninth internatonal conference on Data mining, published by IEEE Computer Society.Scotia, Canada