Incorporating Semantics-based Search and Policy-based Access Control Mechanism in Context Service Delivery

Maria Riaz, Saad Liaquat Kiani, Sungyoung Lee, Young-Koo Lee
Real Time & Multimedia Lab, Kyung Hee University, Giheung, Yongin, Gyeonggi-Do, 449-701, South Korea
{Maria, Saad, Sylee}@oslab.khu.ac.kr, Yklee@.khu.ac.kr

Abstract

Context aware computing utilizes the information embedded in physical and computational environments to provide services adaptive to users' preferences. As these services become more pervasive, the need for a service delivery mechanism arises. Due to the diversity, magnitude and nature of the contextual information, a context-based service delivery mechanism is required which can not only identify the most appropriately matching services for the interested clients but can also ensure that the contextual information is not compromised by providing services to illicit consumers. This paper focuses on the issues of service delivery in pervasive computing environments where services and clients are vast, varied and completely decoupled from each other. Semantics-based attributes are used for service registration and lookup. Privacy is maintained by incorporating policy based access control mechanisms in the service delivery modules.[1]

1. Introduction

The physical space around us is increasingly getting filled with computing elements and smart artifacts that contain a microprocessor. This trend will only pace up in time with more appliances and gadgets appearing embedded with computing and networking capabilities. Ubiquitous computing [1] aims at seamlessly integrating the physical and computational world to enhance the user experience by providing services attentive to a user's preferences in a given context. With the plethora of information gathered from the physical and computational environment, there is a need for an efficient delivery mechanism to filter out unrelated information and communicate the relevant contextual information to its respective clients. Context delivery is the final step in the context awareness process starting from gathering raw data from environment and logical sensors, feature extraction, context synthesis and reasoning, and finally delivering context to the interested client or application. It facilitates the context-based interactions by relating the desired contextual information with the interested context consumer. The clients of a context delivery system include context producers (context aware middleware), that produce and advertise the existence of context aware services; and context consumers (context aware applications), that require these services to emanate intelligent and adaptive behavior.

In a generic service provider / consumer environment, the service is published, discovered based on syntactic parameters i.e., the consumer can lookup the service solely basing on syntax queries. In case the service provider and consumer do not use the same syntax, the desired service could not be provided even if it exists. Similarly, the service interfaces are static and known in advance by the interacting clients. Security and access control models for information are also well understood in case of static service provider and consumer interactions. In a context aware system, a vast set of heterogeneous context services and clients are present. New type of applications and contextual information might be added to the system at runtime which should be incorporated in the system without requiring an update in the service interface towards the existing client applications. Similarly, only syntax-based queries might not be able to identify the desired service since the syntactic representations of the 'same concept' might be different at the client and the service. Finally, the context aware system can easily become invasive from pervasive if access to information is not carefully monitored. A pervasive context aware system senses information from the physical and computational environment and infers useful contextual situations in a user-centric manner. This information, more often than not, is very personal to the user and if available to an unauthorized person can be a serious intrusion in the user's privacy and security. Access control to information is as critical as it is challenging in context aware computing.

In order to meet the additional challenges posed by the peculiar nature of context aware systems, traditional service discovery mechanisms are not enough. We identify

[1] This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment)

two main tasks for service delivery in context aware systems:

- To provide a discovery and registration mechanism which can utilize the underlying contextual information's syntax as well as semantics in order to make more intelligent and accurate matchmaking decisions between the clients and services
- To incorporate dynamic and autonomous access-control mechanisms in the context delivery process to ensure privacy and overall integrity of the system

We propose a context-based service delivery mechanism which affords these functionalities in a context aware environment. The proposed approach is implemented by extending the existing service discovery mechanism of Jini [2] with semantics based parameter specification and search along with access control provision over the available services basing on the requesting client's context. Related work is discussed in section 2. Section 3 presents the overview of the underlying context representation mechanism in our framework. Section 4 presents the context delivery architecture proposed by the authors, and discusses the application of our idea with the help of an example scenario. We conclude our discussion in section 5.

2. Related Work

Context delivery is the final step in the context awareness process. In the existing solutions for context awareness, a variety of approaches have been tried to enable coordination among various parties in a distributed manner. A number of middleware solutions use CORBA based coordination and communication [3], [4] which provides dynamic resource discovery but due to its hard-state registration and lack of support for leasing mechanism, the system behaves reactively instead of proactively to the changing system dynamics. Moreover, the context delivery mechanism only deals with the syntactic comparison and does not provide semantic matching capability. Web services and UDDI based approaches [5] provide a platform-independent mechanism for service description, discovery and usage but it limits the flexibility and expressiveness in the service descriptions to simple attributes URL and string comparisons. Currently, efforts are underway to enable semantic matchmaking capabilities in the UDDI repository for web services [6] [7]. In [8], the authors give architecture and implementation regarding ontology based service discovery. They provide the degree of matching basing on whether the service is an exact, partial or no match. While semantic matchmaking enables support for incomplete or semi-structured information, it does not provide access control over the contextual information.

Traditionally RBAC [9] is used in large scale organizations for scalability and managing complexity. However, it can not enforce fine-grained access control in a context aware environment where the roles are not predefined and the access permissions and constraints are dynamic. Recently, the focus has been given to provide specific mechanisms for handling access control in context aware environments. In [10], the RBAC framework is extended to apply constraints over the role-permission relationship. It allows specification of very fine-grained, flexible, content and time-based access control policies but this approach tightly couples identity/role of client users with their permissions. In [11], two types of monitoring agents keep track of the users' and resources' context respectively and dynamically modify the user-role and resource-permission relationship with the changing contextual situation. However, a mechanism is required to ensure that the consistency is not lost if the user's role is changed during a session or transition. In [12], access control is maintained basing on the current context of the user. This mechanism is good for closed environments such as a meeting or office scenario. However, a user can gain certain privileges by merely entering a specific context, which might not always be desirable.

In this paper, a service delivery mechanism is discussed which enables semantics based discovery and access control in a context aware environments. Our approach differs from the conventional policy-based access control mechanism in that the policies are dynamic and change with the changing system and user's context. This enables the system to enforce access control basing on the current situation. The details are given in section 4.

3. Context Representation in the Framework

Context delivery services, presented in this paper, are built on top of a middleware framework developed by the authors [13]. We represent contextual information using Ontologies [14], which not only allows representation of available contextual information, but also facilitates expressing relationships between different concepts. They enable sharing of similar understanding between various entities in a context aware system. Since, ubiquitous computing environment is characterized by various domains e.g. home, office, university etc; ontology can play a useful role in sharing the domain knowledge of a particular environment.

OWL is a knowledge representation language and has explicit semantics associated with the knowledge. It can be utilized to associate semantic attributes with the context provisioning services at the time of service registration. This overcomes the problems which arise when the clients and services use different syntax for same semantic concepts. The context delivery mechanism utilizes the underlying ontologies to recognize the semantics

associated with the contextual information. For example, from the location ontologies, we can see the *Home* is a type of *IndoorPlace* which is a type of *Place*. This organization helps in identifying the concepts associated with the information and helps in service matching. Figure 1 shows the main context categories and few domain concepts of our context model.

The Semantic Web as a whole is largely conceived as a completely open system, in which everything is published for everyone to see [15]. While the context representation scheme enables semantics-based matching of services, it can not be exploited to enforce access control over the information contents. The context delivery mechanism fills this void by incorporating dynamic context-based policies at the services level as well as at the system level on the whole.

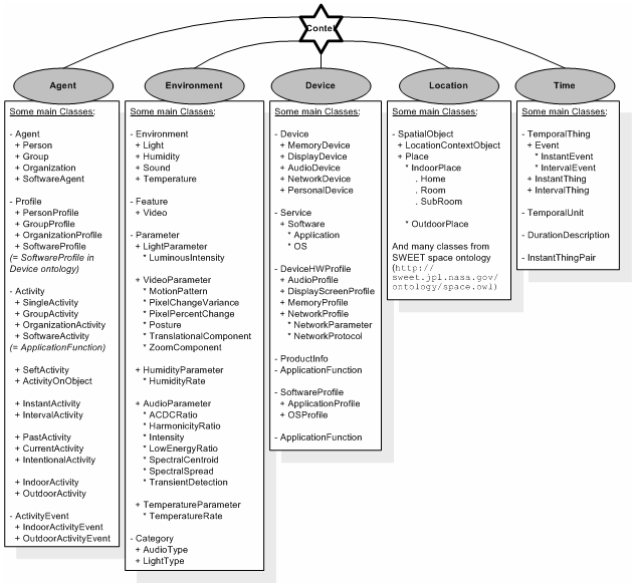

Figure 1. Basic categorization and some domain concepts

4. Context Delivery Services

The context delivery services are present at the top most layer of our framework as shown in figure 2. In the framework, *Feature Extraction* process gathers environmental data and converts it into standardized features. *Ontology Repository* contains the static ontologies and runtime context data in the system. *Context Aggregators* utilize various *Reasoning Modules* to form composite or high-level context from the simple contextual information. Details of each module can be found in [13].

Context Delivery Services provide discovery and lookup functionality to locate appropriate aggregator services and deliver them to the client applications in accordance with the access control policies. Context aggregators register with the registration service to announce information about the context they can provide. This is done by utilizing the concepts defined in the ontology repository. Applications and agents query the registration service for the context of their interest. The registration service, upon finding the appropriate context providers/aggregators, returns their handles to the clients.

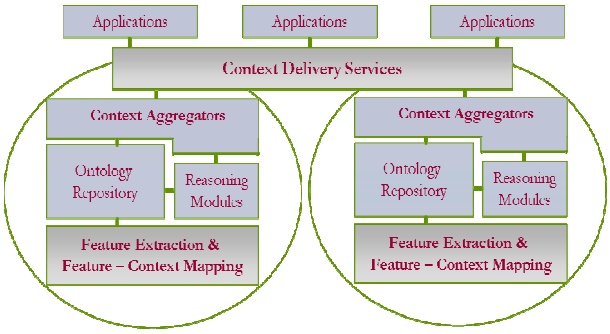

Figure 2. Services in context aware framework

The presence of context delivery services at the user interface level makes them a logical choice for the implementation of access control mechanisms.

4.1. Architecture

The major concerns, in order to define an access control mechanism suitable for pervasive environments, are: policies need to be dynamic in nature; the granularity of control to information needs to be identified; need to cope with changing policies basing on the context at run-time; minimum information inflow to operate and interact with the system in a secure manner.

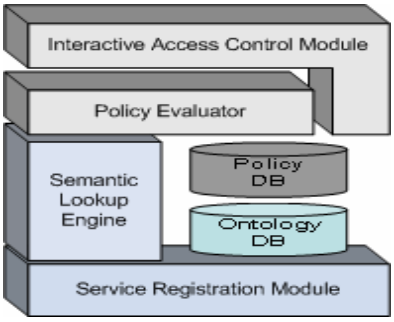

Figure 3. Service delivery architecture

Some of these issues discussed above can be overcome if autonomic access control techniques are employed in conjunction with policies whose applicability can be determined by the current context. The architecture for our context delivery services is shown in figure 3.

Service Registration Module contains the basic functionality of the Jini lookup service. It is the service registry for all the available context aggregator services in the system. The main function of this module is to provide service registration, lookup, event management, and leasing management. Event mechanism enables proactive service discovery notifications, and leasing enables soft state registration and filtering out of stale or unavailable services. Context services and context clients can both locate the service registration module using either unicast or multicast messages. A service registers by specifying a remote reference to the service handle and its semantic attributes basing on the contextual information it provides. The attributes can be service name, type, location, contextual information provided by the service, semantic structure of the information. Other functional attributes can also be associated with the services to make the search more specific. When a client wants to lookup the service, it provides a subset of these attributes and the service which fulfills the search criteria is matched with the lookup request.

Semantic Lookup Engine incorporates matchmaking capabilities to enable service discovery based on the semantic attributes provided in service registration and lookup requests. It utilizes the underlying structure and ontologies of the context data to come up with the most appropriate set of context aggregators to fulfill the applications' contextual requirements. ***Ontology DB*** contains the domain specific categorization of the contextual information. This helps in locating the services that provide the semantic information the client is looking for, even when the syntactic representation of the information at the client and service is different. For example, if a client requests a *Temperature* service, and the system contains a *Weather* service, the semantics will help identify that *Temperature* is a specialization of *Weather* information. Similarly the request can be matched to an *Environment* service since *Temperature* is an attribute of the environment. Once the services are matched according to client's request, the result is given to the *Policy Evaluator* to compute the associated access constraints.

Policy DB contains the system level policies as well as optional aggregator (context provider) level policies, defining the requirements/conditions to access a specific service provided by the context aware system. The policies can be specified by the administrator or service provider and can be updated, modified with changing context.

Policy Evaluator module finds the policies that apply to a specific service in a given context. The function of the policy evaluator can be either proactive or on-demand. In case of proactive behavior, whenever the context changes, evaluator computes the corresponding policy that is now applicable to a service. This is useful if there are many clients which lookup the context services. In case the search queries are not very frequent, on-demand evaluation can be used which computes current applicable policy associated with a service at the arrival of client's request.

Interactive Access Control Module (IACM) interacts with the context clients to obtain credentials required to access a specific service or set of services with associated policies. IACM implements the following operations, discussed in [16], to help in the interactive access control process:

- ***Deduction:*** Given a defined policy and a set of credentials provided by the client, it is decided whether to grant the access or not. It checks if a request for access can be granted or not.
- ***Abduction:*** Given a defined policy and a request to access some resource or service, it is decided what minimum credentials are required so that the given request can be granted. This process is specifically useful in context aware environments where the client might not know what credentials are needed to access a specific service in advance. In that case, IACM will request the client to provide the missing credentials and if the client provides valid credentials, the request is granted, else denied.
- ***Induction:*** Induction utilizes a heuristic function along with some positive examples of scenarios in which the request should be granted and some negative examples of scenarios where the request should be denied. Basing on this information, the induction process tries to identify the policy that satisfies the validity of the granted requests. The induction process is useful in the case where a single static policy can not be defined. This is true for context aware systems since the context changes at run-time, so should the policy to incorporate the new situation.

If the client provides some credentials in the service lookup request, *Deduction* process is used to evaluate whether the credentials satisfy the policy requirements. If the request can not be granted with the provided credentials, or if no credentials were provided at all, *Abduction* process is used to compute the missing credentials. If the client can provide the missing credentials, the request is granted and the services are made available, else the requested service is not provided to the client. *Induction* is used when no clear policy can be computed by the *Policy Evaluator* according to the given contextual situation. The results of induction process are reflected in the *Policy DB* as a positive or negative example for future interactions. In case the user wants to access basic services that do not require any access control, the request is simply granted by returning the list

of most appropriately matched services found by the *Semantic Lookup Engine*.

When a decision to allow access to a certain resource has to be evaluated, the system first checks the current context of the requesting user and the system itself. Basing on the context information, the system evaluates which policy is applicable in the given scenario. Once the current system policy has been identified, the service access request can be evaluated.

In the existing approaches for access control, the policies are mostly static which can not be modified at run-time to incorporate changing system privacy requirements in a pervasive environment. In [17], the authors extend the existing access control model by incorporating roles and delegation mechanism but they only focus on controlling access to devices such as printers and do not define access control over information which is more critical in a pervasive environment. In [18], the authors argue that the access control should be at the information level since the major concern is to maintain the privacy of the contextual information. They provide architecture to maintain access control in a distributed fashion. However, the access control mechanism is not interactive and autonomous. By utilizing the context-based policy selection and the three access control operations defined above, a dynamic and interactive access control mechanism can be put to effect.

4.2. An Example Scenario

The proposed approach can be easily understood by considering a common scenario that can arise in a ubiquitous computing environment, a market place. Take the example of a user who enters a market place to buy some goods. Context aware components present in the market environment can communicate with the user's handheld device to retrieve the list of goods the user needs to buy. Since any person can enter a market place, the user is given the default credentials of '*Customer*', which has limited access privileges.

According to the policy specification, a *Customer* can access services providing information of the available products in the market place but it can not access the accounting and inventory records. The user can conveniently find the goods it requires, check their quantity and price information, pay the bills and accomplish his task. Since the credentials required for this interaction were those of a *Customer*, which were assigned to the user by default, the user does not need to provide any additional information for utilizing the products-related services.

Now consider the case when the user tries to locate the inventory services. The system will find the matching services and compute the policy associated with them, which in this case will be to allow access only to a '*Manager*'. Since the credentials provided by the user (default credentials, *Customer*) do not match the policy, the system will utilize *Abduction* mechanism to compute the minimum required credentials for service access and prompt the user. If the user was indeed the *Manager*, he will be able to provide the missing credentials and the services will be made available to him/her. In case the user can not prove to be the *Manager*, he will be denied access to the inventory records and accounting services. This scenario is depicted in figure 4.

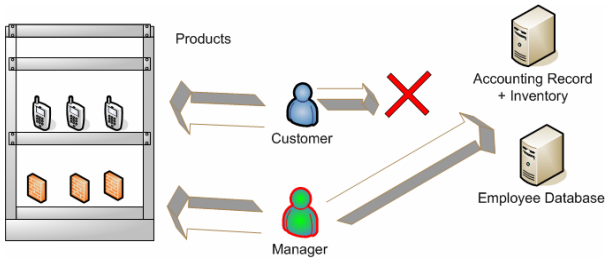

Figure 4. A marketplace scenario

If the user is new to this market place and he doesn't know the exact names of the products he/she is looking for, the semantics based search will enable the user to locate the most appropriately matching services existing in a place new to the user.

Now consider the case when the marketplace closes at night. According to the policy specification, during day time, the default credential for a person entering the market was *Customer*. Since the contextual information has changed, i.e., the market is now closed, the policy related to assigning default credential to the user will also be changed and any user entering now will be assigned the '*Intruder*' credentials. If no exact policy is defined for the detection of intrusion, previous examples can be used to determine what course of action needs to be taken through the *Induction* mechanism. For instance if the last time an intruder was detected, system generated an alarm, then this time again the alarm service will be activated.

For the implementation of the proposed approach, we have extended the service delivery mechanism of Jini. The policies are specified using a java-based policy specification platform. The access control functions are light weight and the algorithm is clear and simple so the overhead caused by enhancing the discovery process is minimized.

5. Conclusion

In this paper, we have provided semantics-based service matching and dynamic policy-based access control mechanisms to improve the service search and lookup functionality and provide privacy over sensitive information and services. The result is a comprehensive

delivery mechanism with effective matchmaking capabilities and reasonable privacy. As a specific case, Jini Network Technology's discovery and registration mechanism is enhanced to incorporate interactive access control and semantics-based search mechanism.

Incorporating access control requires additional information inflow on behalf of the applications i.e., the applications are required to provide some credentials to match the policies. This process might be slow in case some mobile user just wants to retrieve general information e.g., weather, light conditions, humidity, goods available in the market etc, from the context aware system which is not subjected to privacy constraints. In such scenarios, policies can be written to grant unhindered access to services that provide such contextual information. Similarly, incorporating ontologies and semantics in service search poses additional computation burden but given the better hit rate of semantics based matching and high computational capabilities of today's computers, it is not a major concern. OWL is used to define semantics for the information content of the middleware. For access control, various information description languages such as Policy Description Language, XML Access Control Language [19] are present. Currently survey is being done to find the most suitable description language for our purpose.

References

[1] Weiser, M., "The Computer for the 21st Century", Scientific America, Sept. 1991, pp. 94-104; reprinted in IEEE Pervasive Computing, 2002, pp. 19-25.

[2] Sun Microsystems, Inc.: Jini™ Architecture specification. http://www.sun.com/jini/specs/

[3] Román, M., et al., "Gaia: A Middleware Infrastructure to Enable Active Spaces", IEEE Pervasive Computing, Oct-Dec 2002, pp. 74-83.

[4] Yau, S. S., et al., "Reconfigurable Context-Sensitive Middleware for Pervasive Computing", IEEE Pervasive Computing, joint special issue with IEEE Personal Communications, 1(3), July-September 2002, pp.33-40.

[5] Keidl, M., Kemper, A., "Towards Context Aware Adaptable Web Services", Proceedings of the 13th World Wide Web Conference (WWW), New York, USA, May 17-22, 2004, pp. 55-65.

[6] Sheshagirim, M., Sadeh, N. M., Gandon, F., "Using Semantic Web Services for Context-Aware Mobile Applications", MobiSys 2004 Workshop on Context Awareness, Massachusetts, USA, Jun. 2004.

[7] Pokraev, S., Koolwaaij, J., Wibbels, M, "Extending UDDI with context-aware features based on semantic service descriptions", International Conference on Web Services (ICWS), 2003

[8] Broens, T., Pokraev, S., Sinderen, M., Koolwaaij, J., Costa, P. D., "Context-aware, ontology-based, service discovery", EUSAI 2004, Springer, 2004, pp. 72–83.

[9] Giuri, L., Iglio, P., "Role templates for content-based access control", Proceedings of the Second ACM Role-Based Access Control Workshop, November 1997

[10] Tzelepi, S. K., Koukopoulos, D. K., Pangalos, G., "A flexible Content and context-based Access Control Model for Multimedia Medical Image Database Systems", ACM Special Interest Group on Multimedia (SIGMM), 2001

[11] Zhang, G., Parashar, M., "Context-aware Dynamic Access Control for Pervasive Applications", Fourth International Workshop on Grid Computing, Phoenix, Arizona, November 17 - 17, 2003, pp. 101.

[12] Corradi, A., Montanari, R., Tibaldi, D., "Context-based Access Control for Ubiquitous Service Provisioning", 28th Annual International Computer Software and Applications Conference (COMPSAC'04), September 28 - 30, 2004, pp 444-451.

[13] Ngo, H.Q., Shehzad, A., Kiani, S.L., Riaz, M., Lee, S.Y., "Developing Context-Aware Ubiquitous Computing Systems with a Unified Middleware Framework", Proceedings of Embedded and Ubiquitous Computing: EUC 2004, LNCS Volume 3207, Springer-Verlag 2004, pp. 672 – 681.

[14] W3C Web Ontology Working Group: The Web Ontology language: OWL. http://www.w3.org/2001/sw/WebOnt/

[15] Ranganathan, A., McGrath, R. E., Campbell, R. H., Mickunas, M. D., "Use of Ontologies in a Pervasive Computing Environment", The Knowledge Engineering Review, vol. 18, no.3, Cambridge University Press, 2004, pp. 209-220.

[16] Koshutanski, H., Massacci, F., "Deduction, Abduction and Induction, the Reasoning Services for Access Control in Autonomic Communication", proceedings of the 1st IFIP TC6 WG6.6 International Workshop on Autonomic Communication (WAC 2004), Berlin, Germany. Springer, October 2004.

[17] Kagal, T., Finin, L., Josh, A., "Trust-Based Security in Pervasive Computing Environments", IEEE Computer, Dec. 2001, pp. 154–157.

[18] Hengartner, U., Steenkiste, P., "Acccess Control to Information in Pervasive Computing Environments", 9th Workshop on Hot Topics in Operating Systems (HotOS IX), 2003.

[19] http://xml.coverpages.org/xacl.html