Handling Context in a Semantic-based Access Control Framework*

Moussa A. Ehsan, Morteza Amini, Rasool Jalili

Sharif Network Security Center
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
{a_ehsan@ce., m_amini@ce., jalili@}sharif.edu

***Abstract*—As semantic web grows, security concerns increase. One concern is controlling accesses to resources in this environment. In order to infer whether the access is allowed or not, different information of different entities including contextual information should be involved. From access control point of view, we divide the entities in semantic web into three categories: resources (objects), requesters (subjects), and environment (infrastructure, time, and location). In this paper, we present a semantic-based context-aware access control framework to be applied in semantic web, considered as a multi-domain environment. To handle context information in the framework, we propose a context ontology to represent contextual information and employ it in the inference engine. The proposed ontology classifies the context of a semantic web environment and represents the elements of contextual information and their relationship in an abstract level. We illustrate how the access control framework handles the contextual information with the proposed context ontology.**

I. Introduction

Security concerns in distributed environments such as web, and more specifically semantic web, magnify more than other environments. In contrast to traditional environments, where access control is done based upon the identity of users in most of the systems, in distributed environments, other characteristics and information may be employed in access control decision-making process. In a semantic environment, these characteristics and information are semantically related. Such semantically related information may be on account of the relationship between different entities or their instances. Using such information in access control, results in semantic-based access control.

Semantic-based access control focuses on the credentials and other environmental information of the requester to control its access to a resource. Nowadays, several researchers work on different aspects of semantic-based access control models such as security policy languages, but investigation on designing mechanisms of such models are rarely done. In this paper, we focus more on designing security mechanisms in a semantic-based access control framework and more specifically context handling.

*This paper is partially supported by Iran Telecommunication Research Center (ITRC).

Contextual information of the environment is one sort of the important information, which impacts decision making in access control. Context is any information with which the state of an entity can be identified [1]. Several frameworks have been recently proposed to handle context in different environments, but no important one has focused on the sources of contextual information in semantic web. In this paper, we categorize the semantic web into three main entities, which produce contextual information. These include resources (objects), requesters (subjects), and environment (infrastructure, time, and location). Resources are the entities, to be accessed by the requesters. All other means, which let the access to the resource be done, is considered as Infrastructure. The infrastructure, in addition to Time, and Location composes the environment of semantic web. We consider the environment of semantic web as context, and classify it with an ontology to be handled by the framework.

The rest of this paper is organized as follows: The next section surveys the related work. Section III illustrates access control in semantic web, as well as our proposed framework for semantic-based access control. In section IV, the suggested context model for semantic web and the proposed ontology is described in detail. Section V is devoted to the requirements that a semantic-based context handler should meet, and our proposed context handler that fulfills the requirements is discussed. Finally, in the last section we conclude the paper.

II. Related Work

The related work is reviewed in three dimensions: semantic-based access control, context ontology and classifying context in semantic environments, and proposed context handler frameworks.

Most of works done in recent years, are based on the relationships between the entities and available instances in the decentralized environment. Such relationships are usually represented by ontologies and developed with some popular languages such as OWL and RDF.

In another publication, we proposed the semantic-based access control (SBAC) [2], [3] for semantic web. The model uses ontologies as the basis and attempts to reduce semantic relations into the subsumption relations and use them in the

access control. In 2008, we extended the model to support temporal and history-based conditions[4]. The access control has been evolved and logic has been used as the core of inference. A logic-based language named MA(DL)^2 was introduced and used for specifying security policies and reasoning over the semantic information[5]. This model has been applied to our framework [6] in 2008.

There are also many different classifications and ontologies proposed for context in different environments; however, few of them are related to semantic web. Dey *et al.* [7], divided contextual entities into three categories: Place (e.g. a room), People (the individuals), and Things (e.g. physical objects). On the other hand, Bontas *et al.* [8], whose work is the most related work for semantic web, believed that all entities in semantic web are sources of context and can be divided into three categories: Owners, Users, and Environment. Owners refer to all information about the resources and their owners; Users represent the information of user profiles, and Environment contains all other physical and environmental information. However, Bontas *et al.* did not clarify what the definition of the environment especially in semantic web is. We will discuss about this matter in section IV in more detail.

Several frameworks have been also proposed for context handling in different environments, most of them are similar. Dey *et al.* [7] proposed *Context Toolkit* to handle context in a context-aware system. Another framework was introduced by Most'efaoui *et al.* [9] that was designed for security purposes. *Context Broker* proposed by Chen *et al.* [10], [11] is another sample of attempts performed to handle the context in a semantic environment such as a smart room. Most published semantic context-aware systems use ontologies for context modeling. Such ontologies are mostly presented in two levels: a General Level, in which most general and high-level classes are defined; and a Specific Level, in which concentrates in domain-specific classes are specified and usually it is left open for the developers to extend their ontology. Some examples of these ontologies are *SOUPA* [12], its context extension *Cobra-Ont* [13], and *CONON* [14]. On the other hand, some others developed ontologies in only one level such as Strimpako *et al.* [15] and Bontas *et al.* [8]. W3C has also published a working draft on *Delivery Context Ontology* [16], which provides a formal model of what the characteristics of devices in a network are.

III. Access Control in Semantic Web

In order to control accesses in decentralized manner, we divide semantic web into some security domains. In each domain, a security authority and a security agent exists[5]. The authority of the domain is to specify the security policies in its domain; and the security agent is responsible to enforce the specified policies and control the accesses to the registered resources in the domain. In this new definition of semantic web, we assume that an owner registers a resource in one or more security domains and only authorized users may have access to them. The proposed definition for semantic web is shown in Fig. 1.

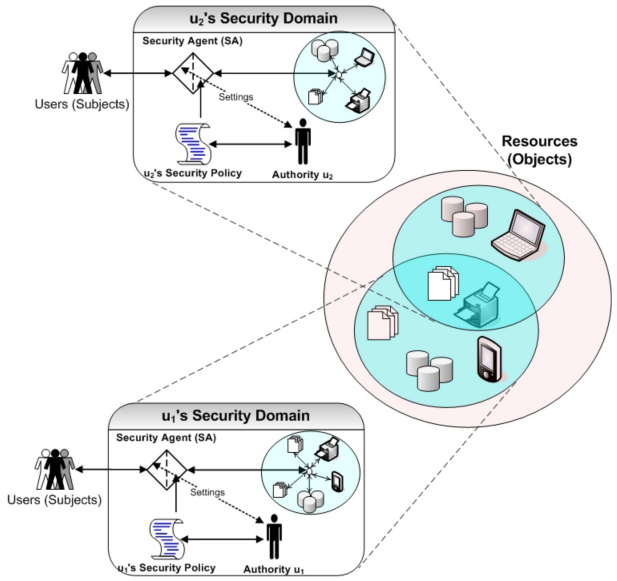

Fig. 1: The Proposed Definition for Semantic Web.

The resources may be annotated by the authority. These annotations contain some information about the security requirements and capabilities that the annotated resource has, and should be satisfied [6]. The access control is done by reasoning over different information of the involved entities in the access request. These entities include the requester, resource, and other environmental (i.e. contextual) entities. We proposed MA(DL)^2 as a logic-based policy language[5], which can be used to specify and infer security policies in semantic-aware environments such as semantic web. We also suggested an ontology to represent the policies based on it [6]. In addition, we propose another ontology to classify the contextual information of this environment, in this paper.

Our proposed security agent framework is shown in Fig. 2. To describe it in short, we illustrate the basis of the framework through an access control scenario. In this scenario, we assume that a credential is verified and validated by *Credential Verifier*. We also assume that all needed information is available in KB. In the following scenario, our domain is an Online Digital Library, which some e-papers and e-books are its registered resources. Considering the aforementioned assumptions, we follow up the scenario of downloading a paper from an Online Digital Library in semantic web:

1) The security authority has defined a policy that only a member can download a paper if its download speed rate is more than 4kb/s.
2) The requester "A" wants to download a paper; therefore, he gives its request to PEP.
3) PEP passes the request to KBMS to extract the necessary information about the paper. The credentials needed in order to show that A is a member is also extracted and all is forwarded to PDP. KB returns the requester download speed rate gathered by the context handler too. However, it may also return the history of access the requester "A"

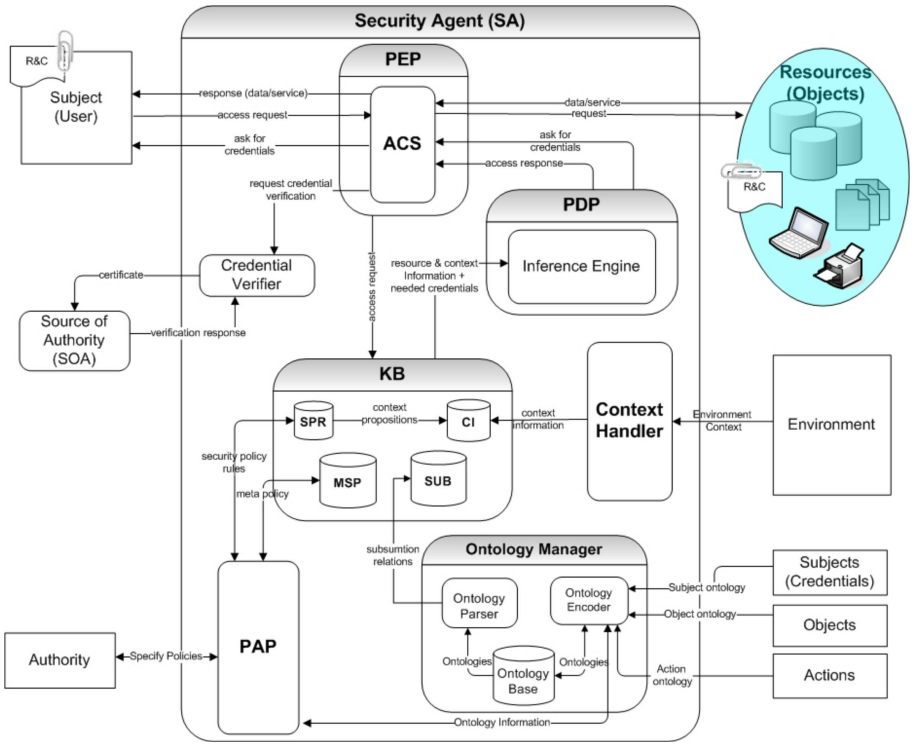

Fig. 2: The Security Agent Framework.

had before.

4) PDP asks for the needed credentials from PEP, and therefore PEP asks for the credentials from "A", and validates them by communicating with Credential Verifier. At last the credentials are passed to PDP.
5) PDP first runs a matching algorithm to see if sufficient credentials are provided [6]. Then, Inference Engine starts reasoning over the available information. If any missing information such as credentials, or other information including context appears, it asks them respectively from PEP, or KB.
6) The decision made by PDP is passed to PEP and if it is admissible, the access is allowed otherwise it is denied. Being allowed to access the paper means that the requester is able to download it.

We will follow the scenario on how the context handler provides the contextual information in section V.

IV. Modeling Context in Semantic Web

In order to use the context information in the semantic-based access control, we first need to find out what the semantic web context consists of. In the model shown in Fig. 3, we claim that semantic web consists of three main entities: Resources, Requesters, and Infrastructure. The resources are the entities located in semantic web and accessed. The requesters are the users or their agents requiring access the resources. The infrastructure is all hardware and software, which are intermediates to let a requester access a resource; therefore, it contains the network on which the semantic web is based.

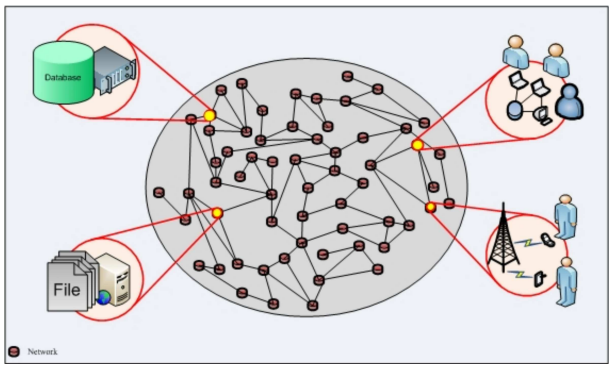

Fig. 3: Categorization of Semantic Web Entities.

The infrastructure ranges between the electronic device that is used as a gate for the requester to connect to semantic web and the server in which the resource is located.

All the three aforementioned entities may be the sources of access control information. Because most information of the requesters and resources are associated with respectively *Subjects* and *Objects* Ontologies (that are defined in Ontology Base), we only refer to context as the infrastructure and environmental information of the requesters and resources (e.g. the time of access request) and let others to be specified in their own scope (defined in madl2 schema [6]). Context is classified with *NSCContext* ontology, which is described in the rest of the section.

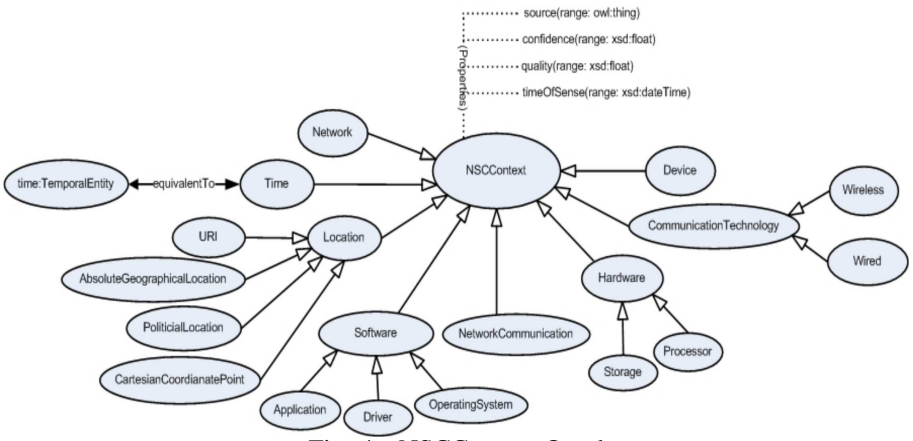

Fig. 4: NSCContext Ontology.

A. NSCContext Ontology

We developed an ontology in OWL[1] by the use of Protege[2]. The ontology categorizes the context of semantic web into eight main categories: *Time*, *Location*, *Network*, *Hardware*, *Software*, *NetworkConnection*, *CommunicationTechnology*, and *Device*. *NSCContext* is shown in Fig. 4. Each piece of contextual information can be defined as an instance *NSCContext* or one of its subclasses. Several properties and subclasses for the context are defined, but if more properties or subclasses are needed, it may be defined due to the ontology flexibility.

The root of the ontology, as depicted in Fig. 4, is *NSCContext*. Each piece of the context information is a subclass of the root and contains all its properties. *NSCContext* properties include *timeOfSense*, *confidence*, and *quality*, and *source*. The three first properties are subclasses of *owl:DatatypeProperty* and the last one is a subclass of *owl:ObjectProperty*. The property *timeOfSense* shows the time that the contextual information is obtained, and its range is *xsd:dataTime*. The properties *confidence* and *quality* show the amount of reliability and accuracy of the obtained piece of information, respectively. The first property is mandatory, but the two last ones are optional. Some mechanisms may be defined to use these two last properties by PDP and KB. The source is the sensor or other entities that senses the context. In the rest of the section, we describe the *NSCContext* ontology.

NSCContext, as aforesaid, consists of eight subclasses. *Time* class is defined equivalent to *time:TemporalEntity* from W3C's standard ontology of Time [17]. The class *Location* has four subclasses including *URI*, *CartesianCoordinationPoint*, *AbsoluteGeographicalLocation*, and *PoliticalLocation*. The class *URI* represents the Universe Resource Identifier in semantic web. *CartesianCoordinationPoint* is a class of a relative point in a defined *CartesianCoordinationSystem*. The other class named *AbsoluteGeographicalLocation* is used for representing location in an absolute manner by the measures: *altitude*, *latitude*, and *longitude*. The class *PoliticalLocation* is to represent the location by the help of *City*, *Country*, and the *Continent*.

The rest of the classes are related to the infrastructure of semantic web. The physical environment of semantic web contains all the entities of the network on what it is built. The main class which may be defined is the *Network*. The *Network* class, as being shown in Fig. 4, is defined by the help of some sub networks. Each network may itself contain some other connections. A *NetworkConnection* is described by a *CommunicationTechnology* between two or more Devices. The *Device* class includes the properties: *communicationTechnology*, *dateOfProduct*, *hardware*, *software*, *manufacturer*, and *model*. The *communicationTechnology* in the range of *CommunicationTechnology*, represents the technology by which the device can communicate with other devices. This class is constructed of two disjoint subclasses: *Wired*, and *Wireless*. The other properties are to demonstrate a device's general information. The properties of *Hardware* and *Software* classes are similar to *Device*. *Hardware* has two disjoint subclasses: *Processor* and *Storage*. Three subclasses of *Software* are defined as *Application*, *Driver*, and *OperatingSystem*.

V. Context Handling in Semantic Web

By defining an ontology, controlling the access to resources is facilitated. Now the framework should provide some mechanisms to obtain the context from the environment, check the quality of the obtained piece of information, store it in KB, and convert the contextual piece of information to a suitable format or accuracy needed for a specific decision making. In the next subsection, some requirements that a framework should have for handling context is listed. Then the proposed context handler to be used in the access control framework is described.

A. Requirements for a Context Handler

Surveying the other proposed context handlers in the literature and investigating decentralized access control frameworks,

[1] http://www.w3.org/TR/owl-features/
[2] http://protege.stanford.edu/

the essential requirements for a suitable context handler are as follows:

1) Diversity of retrieving and handling context [11]. The types of context are different; therefore, the types of retrieving and handling the contextual information are different too. The proposed context handler should be able to work and communicate with variant devices that sense the contextual information.
2) Interpretation of context [7]. Due to the variety of the usage of contextual information, different interpretation is assumed. Therefore, the framework should be able to interpret the information for a specific usage.
3) Transparency of distributed communications [7]. Input of data in traditional environments has been through using some traditional devices such as local keyboard or mouse. In handling context, data is coming from different devices and from different places.
4) Full availability in retrieving context [7]. Some contextual information may be retrieved on demand. Some others may be sensed not at the time of need, stored and used later. Therefore, the context handler should be able to accept the data from different devices all the time, and store them for future usage.
5) Contextual information quality [18]. Some contextual information may be obtained from different sensors and different locations; e.g. the location of the requester may be obtained from a RFID or GPS. The quality of these devices may not be the same. So some methods should be defined to extract the worth information.
6) Context repository and maintaining the history [9]. Contextual data should be stored in a repository and maintained for future usage. For example the movement of a user may be tracked.

B. A Framework of the Context Handler

We propose a context handler to be used in the semantic-based access control framework. In the framework, most of the aforementioned requirements are being considered. The context handler framework is shown in Fig. 5. It is related to two other external entities: KB, and Environment. KB is a repository in which the contextual information is stored. It asks for contextual information from Context Handler, and interprets it too. The Context Handler gets the contextual information from the Environment. Some devices are located in the environment, which are responsible for sensing the contextual information.

The Context Handler contains three main parts. Context Interface is an interface from which KB may ask for contextual information. The Context Acquisition is responsible to obtain the sensed information from the environment. The acquisition may be done after receiving a request or by being triggered from the sensors in the environment. The Context Quality Controller controls the quality of the retrieved information, and lets Context Interface return the sensed information. The sensed pieces of contextual information are stored in the CI.

```
<NetworkConnection rdf:ID="D221">
  <connectionSpeedRate>
    <ConnectionSpeedRateMeasure rdf:ID="F325">
      <unit>
        <ConnectionSpeedRate rdf:ID="Kb_s">
          <measureConversionFactorToBaseMeasure
              rdf:datatype="&xsd;#float">
              1.0
          </measureConversionFactorToBaseMeasure>
          <baseMeasureUnit rdf:resource="#Kb_s"/>
          <measureUnit rdf:datatype="&xsd;#string">
          kb/s
          </measureUnit>
        </ConnectionSpeedRate>
      </unit>
      <amount rdf:datatype="&xsd;#float">3.2</amount>
    </ConnectionSpeedRateMeasure>
  </connectionSpeedRate>
  <source>
    <Client rdf:ID="DU_Meter_4.0">
      <version rdf:datatype="&xsd;#string">4.01</version>
      <source rdf:resource="#Hagel_Technologies_Ltd"/>
      <confidence rdf:datatype="&xsd;#float">1.0</confidence>
      <vendor rdf:resource="#Hagel_Technologies_Ltd"/>
      <quality rdf:datatype="&xsd;#string">1</quality>
    </Client>
  </source>
  <timeOfSense rdf:datatype="&xsd;#dateTime">
    2008-09-15T00:00:00
  </timeOfSense>
  <communicationTechnology rdf:resource="#Cable"/>
  <confidence rdf:datatype="&xsd;#float">1.0</confidence>
</NetworkConnection>
```

Fig. 6. Represented Context Information.

Then if any information is asked from PDP, the Context Interpreter interprets and returns it.

Consider the scenario discussed in section III. PDP needs the download speed rate of the requester's connection to decide about the access; therefore:

1) KB asks Context Handler for the *NetworkConnection* contextual information. The request is given via Context Interface.
2) Context Interface passes the request to Context Acquisition.
3) Context Acquisition may communicate with some sensors, by the use of Physical Environment interface. The methods of how to gather the download speed rate is provided in Context Handler.
4) After achieving the download speed rate, it is passed to the Context Quality Controller.
5) By validating the gathered value by some defined factors, it is passed to the Context Interface.
6) Context Interface returns the value to KB and specifically KBMS.
7) KBMS stores the download speed rate in CI and represents it by the *NSCContext* ontology (see Fig. 6)
8) Finally, the value is provided to PDP to decide about the access request.

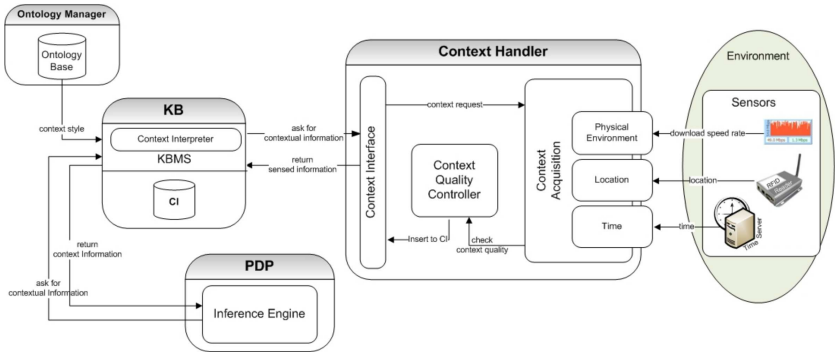

Fig. 5: The Proposed Framework for Context Handler.

C. Discussion

The proposed framework meets most of the requirements aforementioned. The diversity of the context sources is one of the important requirements that should be met. This requirement is met by having Context Acquisition in the context handler which operates as a middleware. The middleware is connected to different sensors that sense the contextual information. We believe that all sensors can be modeled as some servers that are connected to them and the servers should communicate with the middleware. So, different types of context can be handled. Also, Context Acquisition can make all distributed communications transparent, and therefore, meet another requirement.

The contextual information is differently interpreted per usage. This results in another requirement, which is the responsibility of the Context Interpreter in KB. Context Quality Controller is to control the quality of the retrieved information. In the access control framework, CI is a repository to store the contextual information, and the history of the required context. The only lack is how to make the framework available all time. We believe that this requirement is not in the same abstraction level that we are talking about at the moment. On the future work, we are going to implement the framework in a real situation; so, we will offer a solution to meet it at that time.

VI. Conclusions

In this paper, we modeled semantic web as three main entities: resources (objects), requesters (subjects), and environment (infrastructure, time, and location), due to their impact on access control. Furthermore, we developed an ontology to represent the infrastructure and therefore the contextual information in the semantic web environment. In the ontology, all sources of contextual information in that environment are classified. The contextual information represented with the ontology was used in controlling accesses. In addition, some requirements that a semantic context handler should meet was discussed. The paper also proposed a framework that met those aforesaid requirements.

References

[1] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4 – 7, 2001.

[2] S. Javanmardi, M. Amini, and R. Jalili, "An access control model for protecting semantic web resources," in *2nd International Semantic Web Policy Workshop (SWPW'06)*, Athens, GA, USA, 2006, pp. 32–46.

[3] S. Javanmardi, M. Amini, R. Jalili, and Y. GanjiSaffar, "*SBAC*: A semantic based access control model," in *11th Nordic Workshop on Secure IT-systems (NordSec2006)*, Linkping, Sweden, 2006.

[4] A. Noorollahi, M. Amini, and R. Jalili, "A temporal semantic-based access control model," in *13th International CSI Computer Science Conference(CSICC '08)*. Kish Island, Iran: Springer-Verlag, 2008.

[5] M. Amini and R. Jalili, "Specification and inference of authorization and obligation policies using deontic logic for semantic-aware environments," in *4th Iranian Society of Cryptology Conference (ISCC07)*, Tehran, Iran, 2007, pp. 175–184.

[6] M. Ehsan, M. Amini, and R. Jalili, "Using semantic annotation to design security mechanisms based on the semantic-aware access control $MA(DL)^2$," 2008.

[7] A. K. Dey, D. Salber, and G. D. Abowd, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Computer Interaction (HCI) Journal*, vol. 16, no. 2-4, pp. 97–166, 2001.

[8] E. P. Bontas, "Representing context on the semanticweb," pp. 308–313, 2004.

[9] G. K. Mostefaoui and P. Brezillon, "A generic framework for context-based distributed authorizations," pp. 204–217, 2003.

[10] H. Chen, F. Perich, D. Chakraborty, T. Finin, and A. Joshi, "Intelligent agents meet semantic web in a smart meeting room," in *Third International Joint Conference on Autonomous Agents and Multiagent Systems(AAMAS'04)*, vol. 2, New York, NY, USA, 2004, pp. 854–861.

[11] H. Chen, T. Finin, A. Joshi, L. Kagal, F. Perich, and D. Chakraborty, "Intelligent agents meet the semantic web in smart spaces," *IEEE Internet Computing*, vol. 8, no. 6, pp. 69–79, 2004.

[12] H. Chen, F. Perich, T. Finin, and A. Joshi, "*SOUPA*: Standard ontology for ubiquitous and pervasive applications," in *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04)*, Boston, Massachusetts, USA, 2004, pp. 258–267.

[13] H. Chen, T. Finin, and A. Joshi, "An ontology for context-aware pervasive computing environments," *Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review*, vol. 18, no. 3, pp. 197–207, 2004.

[14] T. Gu, X. H. Wang, H. K. Pung, and D. Q. Zhang, "An ontology-based context model in intelligent environments," in *Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS'04)*, San Diego, California, USA, 2004.

[15] M. A. Strimpakou, I. G. Roussaki, and M. E. Anagnostou, "A context ontology for pervasive service provision," in *the 20th International Conference on Advanced Information Networking and Applications (AINA'06)*, vol. 2. Vienna, Austria: IEEE Computer Society, 2006, pp. 775 – 779.

[16] R. Lewis and J. M. C. Fonseca, "Delivery context ontology," 2008.

[17] J. R. Hobbs and F. Pan, "Time ontology in owl," 2006.

[18] D. Preuveneers and Y. Berbers, "Quality extensions and uncertainty handling for context ontologies," in *Workshop on Context and Ontologies: Theory, Practice and Applications*, Riva del Garda, Italy, 2006.