Building Context-Aware Access Control In Enterprise Ontologies

Research-in-Progress

Long Flory
Department of Information Systems
Virginia Commonwealth University
floryl@vcu.edu

ABSTRACT

Knowledge centric management (KCM) has become a key strategy for competitive edge. As an essential of KCM, an enterprise ontology represents the knowledge of an organization. Thus, the need for securing enterprise ontologies (EO) becomes imperative. Adequate access control is a major component of ontology security. However, access control for EO is largely neglected in information systems (IS) literature. This paper presents the first research to fill this gap. I propose five requirements for good access-control solutions for EO. The proposed solution offers an architecture framework that meets the five requirements. Semantic Web technology is used to build context-aware access controls into EO. My proposal includes a novel resolution for policy conflicts. This study provides the first design of fine-grained and dynamically-adjusted access authorizations.

Keywords

Semantic Web technology, enterprise ontology, ontology security, access control, access-control requirement, context-aware, dynamically-adjusted, fine-grained, policy conflict resolution.

INTRODUCTION

Global competition propels businesses towards knowledge centric management in which enterprise ontologies are the key players. An enterprise ontology is a model that represents the knowledge of an organization. EO distinguishes itself from other types of ontologies in the uses. An enterprise ontology typically has predefined users who are granted specific access privileges (Fox, Barbuceanu and Gruninger, 1995; Guinn, 2011; Uschold, King, Moralee and Zorgios, 1998). EO is the pivot in Business Process Engineering, Strategic Planning, and Management Information Systems (Guinn, 2011).

Enterprise ontologies are organizational assets. The need for securing EO becomes more imperative than ever before. The ontology security needs adequate access control. It ensures that authenticated users perform the authorized activities to an enterprise ontology (Chen and Stuckenschmidt, 2012; DISA, 2004; Hu, Ferraiolo and Kuhn, 2006). Unfortunately, access control for EO is largely neglected in IS literature. A few prior works attempt to address the issues of the access control. However, the proposed approaches are inadequate to address the issues in rich ontology inferences (Chen and Stuckenschmidt, 2012; Jain and Farkas, 2006). The access control methods for relational databases are widely available, but they are incapable of handling the policy conflicts raised from the ontology inferences (Reddivari, Finin and Joshi, 2007). The access control for EO **needs emphasis in IS researches**. This study aims to fill this gap in IS literature.

An access-control system performs the *user authentication* in which the system validates the users and the *access authorization* in which the system declines or permits the activities of the users (DISA, 2004; Hu et al., 2006). This study concentrates on the access authorization, the major challenge in the access control. Specifically, I address three research questions:

> *What are the requirements for good access-control solutions for enterprise ontologies*? *How do we build context-aware access-control into enterprise ontologies? What is a good architecture framework for access-control solutions that meet the identified requirements?*

To address the research questions, this paper proposes five requirements for good access-control solutions for EO. I also offer a novel access-control approach that is fine-grained, context-aware, and dynamically-adjusted. Fine-grained means my proposal can control the accesses to any grain of an enterprise ontology according to its particular confidentiality. Context-aware means the access-control system declines or permits an activity according to not only the users' authentications but also the contexts in which the users perform the activities to an enterprise ontology (**Definition 3** of this paper defines the concept of context). Dynamically-adjusted means the system will automatically decline an ongoing activity to the enterprise ontology as soon as a change of the context raises a threat to the enterprise ontology. In the subsequent sections, I will illustrate the importance of the context awareness and the dynamic adjustment. In summary, my proposal builds adequate access control into EO and supports ease-of-maintenance. Additionally, the proposed solution is capable of resolving *policy conflicts* that occur when two access control policies are inconsistent. With the best knowledge I have, this study is the first research proposing **a comprehensive solution**.

The structure of this paper includes the literature review in the next section. Then, I propose the five requirements for good access-control solutions for EO. Next, I build context-aware access control into EO. As followed, I propose the architecture framework and provide a use scenario. Lastly, I discuss the future research and conclude this paper.

RELATED WORKS

IS researchers have recognized the need for the access control for ontologies (Duan, Kementsietsidis, Le and Wang, 2012; Pavlov, 2011). The current approaches mainly adapt two traditional access-control techniques, namely the Role-Based Access Control (RBAC) and the Label-Based Access Control (LBAC). RBAC aims at reducing the workloads in policy maintenances. The technique groups the users sharing the same access privilege into a role. When the role of a user is changed, the ontology administrator simply reassigns a new role to the user and does not need modifying the access control policies. LBAC supports fine-grained access control. The approach labels the data elements with their respective confidentialities. However, few researches address the access control for EO. This section reviews the relevant works.

Query Rewriting

Several researches propose the query-rewriting approaches. They use access-control proxies, software modules to rewrite the user queries according to pre-defined RBAC policies. Some proposal uses OWL reasoner to address *policy conflicts*. To automate the query-rewriting, ontology developers have to pre-define the rules for it. This is an extremely difficult task. Consequently, the use of these approaches is limited. Moreover, query-rewriting tends to suppress useful information because the rules enforce the most restrictive policy on the multiple queries of a user. Chen and Stuckenschmidt (2012) implement a query-rewriting approach. The work rewrites the SPARQL queries in the XACML (Extensible Access Control Markup Language).

Axiom Filtering

Axiom-filtering techniques attempt to reduce information suppressing (Baader, Knechtel and Pe˜naloza, 2009; Calvanese, De Giacomo, Lenzerini and Rosati 2008; Knechtel, 2008). The techniques label ontology components with pre-defined axioms. If an access should be declined, the system answers the user queries with the axioms rather than the requested data. So, the axiom-filtering partially resolves information suppressing problem. Still, ontology developers have to pre-define the axioms. This is also a difficult task. In general, axiom-filtering techniques do not address policy conflicts.

Access controls for the RDF stores

A number of researches propose the access-control solutions for the Resource Description Framework (RDF) stores. One work offers context-aware access control but does not resolve policy conflicts (Abel, De Coi, Henze, Koesling, Krause and Olmedilla, 2007). Reddivari et al. (2007) implement a RDF-Store Access-Control Policies (RAP) to enforce user-defined policies. RAP stores the policies together with the metadata of RDF-triples. A policy engine mediates a user activity and determines its access authorization. However, RAP does not resolve policy conflicts. Jain and Farkas (2006) apply LBAC on RDF-triples. The solution proposes subsumed RDF-patterns to resolve policy conflicts. However, the approach is not context-aware.

Context-aware Access Controls

Zhang and Manish (2003) propose the Context Agent to achieve dynamically-adjusted access control for the pervasive applications. When a user queries the ontology, a set of roles is assigned to the user. But, only one role is active for the user according to the context at a moment. Later, Damiani, Dertino, Catania and Perlasca (2007) use the spatial-contexts to control the active role. In a spatial context, a role is either active or inactive. But, the access authorization is not dynamically-adjusted. Commonly, these two works tend to produce large amounts of specialized roles, and thus, lose the advantage of RBAC. More recently, some researches use the level of trust to control the access to an ontology (G¨otz, 2012; Ma, He and Gao, 2012). This approach has the difficulty to distinguish between trusted and hostile behaviors.

REQUIREMENTS FOR GOOD ACCESS CONTROLS FOR ENTERPRISE ONTOLOGIES

Few IS researches, if any, have discussed the requirements for good access-control solutions for EO. This paper attempts to address them. On the basis of extensive literature review, I adopt the theoretical frameworks of U.S. Department of Defense (DISA, 2004) and U.S. National Institute of Standards and Technology (NIST) (Hu et al., 2006). DISA and NIST define that access controls are the mechanisms to determine the allowed activities of the users and to mediate every attempt to access the resources in an information system. An access-control system needs performing the user authentication and access authorization in order to enforce the ***access control policies,*** the organizational requirements for governing the accesses to the resources in the information systems. DISA and NIST have standardized their respective frameworks. They provide the principles of the access control for information systems. Hereinafter summarizes the principles relevant to the access control for EO.

1. Providing fine-grained access controls: Securing any grain according to its specific confidentiality.
2. Supporting *Separation of Duty (SOD)*: No user should be given enough access privilege to misuse the system.
3. Offering means to resolve the policy conflicts.
4. Achieving suitable efficiency: Performance is critical for a system with large concurrent transactions.
5. Minimizing the workloads in maintaining access control policies.

The two frameworks do not highlight context-awareness. However, many prior studies have demonstrated that context-awareness is an important requirement for the access control for ontologies (Abel et al., 2007; Bertino and Kirkpatrick, 2011; Bolchini, Curino, Orsi, Quintarelli, Rossato, Schreiber and Tanca, 2009; Damiani et al., 2007; Zhang and Manish, 2003). Extending the principles of DISA and NIST as well as the previous researches, I propose five requirements for good access-control solutions for EO:

1. *Granularity:* Protecting the specific confidentiality of any grain of an enterprise ontology.
2. *Dynamically-context-aware*: Automatically declining an ongoing access to the enterprise ontology as soon as a change of the context raises a threat to the enterprise ontology.
3. *Separation of Duty (SOD)*: No user is given enough access privilege to misuse the system.
4. *Conflict-proof*: Automatically detecting and resolving policy conflicts.
5. *Ease-of-maintenance*: Enabling minimum workload in maintaining the access control policies.

The requirements of *granularity*, *separation of duty*, and *ease-of-maintenance* are similar to the corresponding principles of DISA and NIST. But, I propose a new requirement, *dynamically-context-aware*. It requires an access-control system grant different access authorizations to the same user (role) in different contexts. This is because an enterprise-ontology is often more vulnerable in one context than another. For instance, an enterprise-ontology is more vulnerable to network hacks when a manager accesses the ontology via a public network than via the company's intranet. Additionally, *dynamically-context-aware* requires an access-control system continue monitoring the context in which a user performs ongoing activities to the enterprise ontology. When a contextual change increases the restriction on the user's activities, the system should automatically enforce the higher stringency. To illustrate, let's consider this scenario: Data analyst Mark is accessing the company's enterprise ontology in his office. While his PC is connecting to the enterprise ontology, Mark leaves his office for a few minutes. In the absence of Mark, his PC opens the doors for unauthorized accesses to the enterprise ontology. Such negligence often occurs in the uses of information systems (DISA, 2004; Hu et al., 2006). If the access-control system is *dynamically-context-aware*, it can permit the access only when Mark stays within a certain range of his PC. The system should automatically logout Mark's PC from the enterprise ontology when Mark is absent from that range.

In addition, automatic *conflict-proof* is especially important for the access control for EO. Police conflicts can occur from the ontological inferences when querying an enterprise ontology. Inferences are not an issue when querying relational databases, XML data repositories, or other file systems. Finally, performance should not be a pressing issue in the access control for EO because there are a small number of concurrent activities in an enterprise-ontology system. It is distinct from a web-service system where there are a large number of concurrent activities. A web service often does not predefine its users whereas an enterprise ontology predefines its users. Different users are usually allowed to access the enterprise ontology in different time periods. In a given period of time, the number of access activities to an enterprise ontology is often small.

To meet the five requirements, I propose building context-aware access controls into EO.

BUILDING CONTEXT-AWARE ACCESS CONTROL INTO ENTERPRISE ONTOLOGIES

This section contains three subsections. The first subsection defines three terms. In the second subsection, I propose the ***access-control properties*** that build context-aware access control into EO. The third subsection proposes the *Resolution Rule for Policy Conflicts*.

Three Terms

Definition 1 (Secured Objects): A secured object is a portion of an enterprise ontology, which has certain confidentiality and requires certain level of access control. A secured object is denoted as SB_g.

A secured object can be any grain of an enterprise-ontology, for example, a class, a property, an instance, a rule, a group of classes, etc.

Definition 2 (Legitimate Roles): A legitimate role is a group of users who share the same access privilege to access an enterprise ontology. A legitimate role is denoted as LR_h.

A legitimate role can be a ***human role*** or an ***artificial role***. *Human roles* may include ontology developer, department assistant, analyst, etc. *Artificial roles* are application programs to which certain access privileges are granted.

For context-awareness, I propose four contextual attributes: ***location***, ***time***, ***application***, ***device*.**

1. ***Location***: The geographical range in which an access to an enterprise-ontology occurs. A *location* has a status. It is either ***in*** or ***out***. Status *in* means the access is permitted at the location. Status *out* means the access is not allowed at the location.
2. ***Time***: The time when an access to an enterprise-ontology occurs. A *time* has a status. It is either ***on*** or ***off***. Status *on* means the access is permitted at the time. Status *off* means that the access is not allowed at the time.
3. ***Application***: The application program used to access the enterprise ontology. An *application* has a status. It is either *alw* or *don*. Status *alw* means the program is allowed to perform the access. Status *don* means the program is not allowed to perform the access.
4. ***Device***: The electronic device used to access the enterprise ontology. A *device* has a status. It is ***asg***, ***cmp***, or ***prv***. Status *asg* means the device is assigned to access the enterprise ontology. Status *cmp* means the device is owned by the organization but not specifically assigned to access the enterprise ontology. Status *prv* means the device is owned by an employee of the organization.

Definition 3 (Contexts): A context is a combination of the statuses of all contextual attributes.

Example-1: According to the four contextual attributes defined above, here are six examples of contexts:

C_1 (in, on, alw, asg)	C_2 (out, on, alw, asg)	C_3 (in, on, alw, cmp)
C_4 (out, on, alw, cmp)	C_5 (in, on, alw, prv)	C_6 (out, on, alw, prv)

Where C_j (j = 1, 2, 3, 4, 5, 6) is the j^{th} context.

Access-Control Properties

Before defining *Access-Control Properties*, I shall stress that access control policies should be written in the terms consistent with those used by the enterprise ontology. Moreover, an access control policy should be a positive policy that expresses *'allow to'* rather than *'not allow to'*.

Example-2: Here are two examples of access control policies:

(1) An analyst can retrieve *Customer* class in work hours and with an allowed application.
(2) An analyst can retrieve *Product* class in office and with assigned computer.

Definition 4 (Access-Control Properties): Let O_k be an operation on EO. O_k can be retrieval, updating, creation, deletion, etc. Let C_j be a context. $ACP_{gh} = (O_k, C_j)$ denotes that legitimate role LR_h is allowed to perform O_k on secured object SB_g in C_j. ACP_{gh} is called an ***access-control property*** of SB_g relative to LR_h.

Intuitively, SB_g can have many *access-control properties* each of which relates SB_g to LR_h or another legitimate role (Figure 1). Thus, the *access-control properties* are ***build-in access control polices*** in EO. Moreover, the *access-control properties* meet the requirements of *granularity* and *context-aware*. SB_g can be any grain of an enterprise-ontology. The access-control properties of SB_g enforce the access control policies with context-awareness (C_j). SB_g, O_k, and C_j minimize the access privilege of LR_h. So, the *access-control properties* meet the requirement of *separation of duty*.

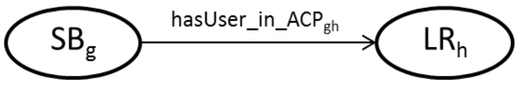

Figure 1. An Access-Control Property

Policy Conflict Resolutions

Policy conflicts often occur when requested objects are involved in subsumption relations. To implement *conflict-proof*, I propose a rule to resolve policy conflicts.

Definition 5 (Resolution Rule for Policy Conflicts): Let ACP_{mh} be an access-control property of secured object SB_m; ACP_{kh} be an access-control property of secured object SB_k. If SB_m subsumes SB_k, then both ACP_{mh} and ACP_{kh} can be applied to SB_k via the ontology inference. If ACP_{mh} is more restrictive than ACP_{kh}, then ACP_{mh} will be applied to SB_k. Otherwise, ACP_{kh} will be applied to SB_k.

Since the subsumed object should be more sensitive than its parent objects, the proposed rule allows the appropriate access control to be enforced. Furthermore, this rule does not suppress the accesses to the secured objects that are not affected by the subsumption relation.

ARCHITECTURE FRAMEWORK

This section proposes an architecture framework to enforce *dynamically-context-aware* access control for EO. Figure 2 depicts the framework including a 4-step workflow.

1. *User Authentication (UA)*: The UA interfaces between the users and the enterprise ontology.
   1) Authenticating the user (e.g. IDs and passwords).
   2) Assigning a legitimate role LR_h to the user. If the user fails the authentication, the system returns an error message to the user and declines the login.
   3) Allowing LR_h to submit queries called ***role queries*** via a query interface (a software module is under design as a part of my ongoing research. The module will automatically process the role queries).
   4) Retrieving the current context and passing on the context, legitimate role information, and role queries to the Access Evaluator.
2. *Access Evaluator (AE)*: The AE receives the information from UA and retrieves the relevant access-control properties. AE evaluates the role queries against the access-control properties. If the role queries pass the evaluation, AE passes on them to the Access Authorization (AA). Otherwise, AE declines the role queries and sends an error message to UA that passes on the message to the user.

   If a policy conflict occurs, AE passes on it to the *Conflicts Manager* (*CM*) that resolves the conflict according to the *Resolution Rule for Policy Conflicts*.
3. *Access Authorization (AA)*: The AA verifies the role queries delivered by AE and stamps a “permit” on the role queries that pass the verification. Then, AA passes on the stamped queries to the *Query Optimization and Processing*. If the verification indicates a problem, AA will return the role queries to AE, which will resolve the problem or decline the role queries.
4. *Query Optimization and Processing*: The ontology query-engine in the enterprise ontology optimizes the role queries, executes the optimized queries, and returns the final results to AE. It checks the current context to see whether the final results should be rejected or passed on to UA. If UA receives some final results from AE, UA will pass on them to the user.

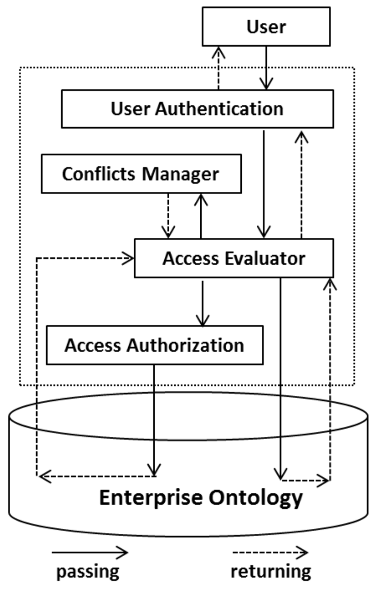

Figure 2. Architecture Framework

In the four steps described above, *dynamically-adjusted context-aware* access control is enforced. UA continues providing the current context to AE as long as the user is still accessing the enterprise ontology. AE will immediately reject a role query or a query result when a context change disallows it. Also, I consider that query optimization should exist in an enterprise-ontology system and does not need to be addressed by the access-control solution. The implementations of UA, AE, CM, and AA are in the agenda of my ongoing research.

A USE SCENARIO

This section provides a use scenario to illustrate the use of the proposed solution. The first phase in using the solution is to define the contexts. Here, I use **Definition 2 and 3** and skip the first phase. Also, I use the *Product Order Ontology* (POO) given in Figure 3. The second phase is to define the legitimate roles.

Defining Legitimate Roles

1. *Artificial role I* (LR$_1$): The computers to which certain access privileges are assigned.
2. *Artificial role II* (LR$_2$): The applications to which certain access privileges are assigned.
3. *Ontology developer* (LR$_3$): This human role designs and implements the POO.
4. *Department assistant* (LR$_4$): This human role supports the department managers.
5. *Marketing analyst* (LR$_5$): This human role analyzes marketing data and supports marketing decision making.
6. *Financial analyst* (LR$_6$): This human role analyzes financial data and supports financial decision making.

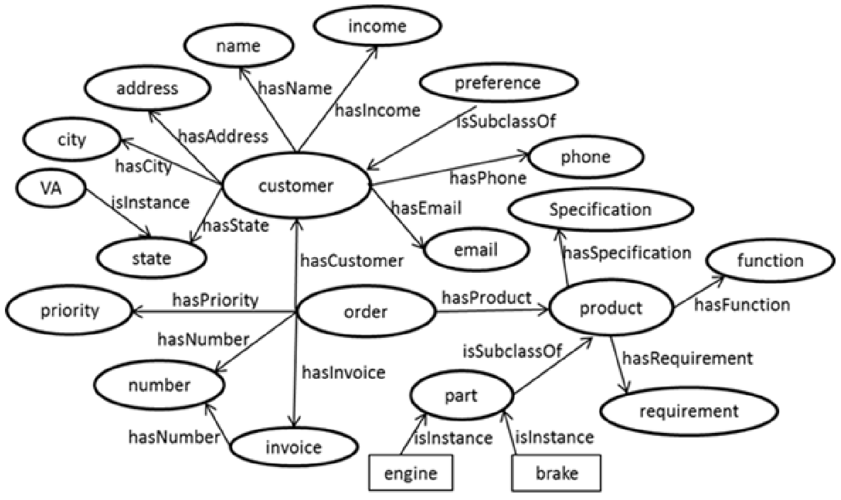

Figure 3. A Product Order Ontology

Defining Access Control Policies

The third phase is to define access control policies. Table 1 provides seven examples of access control policies.

Labels	Access Control Policies
1.	*Artificial Role I* (LR$_1$) is allowed to retrieve customer and product classes only during time period I.
2.	*Artificial Role II* (LR$_2$) is allowed to retrieve customer and product classes only during time period II.
3.	*Ontology developer* (LR$_3$) is allowed to perform creation and clear on the POO when the operations are performed in acceptable offices, in working hours, and by assigned computers and applications.
4.	*Department assistant* (LR$_4$) is allowed to insert related data into the POO when the operations are performed in acceptable offices, in working hours, and by assigned computers and applications.
5.	*Marketing analyst* (LR$_5$) is allowed to retrieve customer class when the operations are performed in working hours and by assigned applications.
6.	*Marketing analyst* (LR$_5$) can retrieve customer preference when the operations are performed in working hours, in acceptable offices, and by assigned applications.
7.	*Financial analyst* (LR$_6$) can retrieve order class when the operations are performed in acceptable offices and by company computers and assigned applications.

Table 1: Examples of Access Control Policies

Building Access-Control Policies in the POO

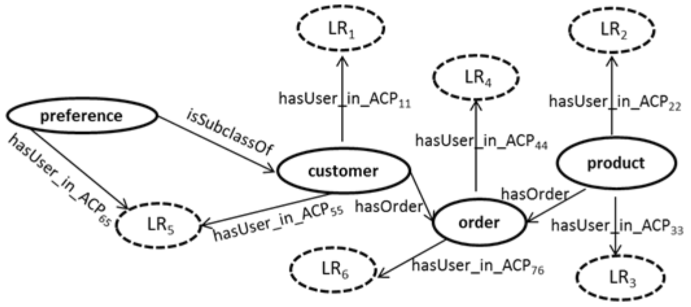

Figure 4. A Portion of the Secured Product Order Ontology

The fourth phase is to define the access-control properties according to the access control policies. A software module is under design as a part of my ongoing research. The module will automatically generate the access-control properties using the access control policies. The module will also automate the maintenances of the access-control properties. In this paper, I use Protégé to accomplish this task. Figure 4 provides a portion of the access-control properties. The graph also indicates that the proposed solution meets the *ease-of-maintenance* requirement. Adding a new access control policy is simply adding new access-control properties. Changing an access control policy is simply changing the affected access-control properties. In the policy maintenances, the unaffected access-control properties are generally intact. After building the access-control properties, an automatic diagnosis runs to check for policy conflicts.

Resolving Policy Conflicts

The fifth phrase in using the proposed solution is to identify and resolve policy conflicts. In Table 1, both fifth and sixth access control policies can be enforced on the secured object *preference*, which has its own access-control property ACP_{65} and also inherits ACP_{55} from the parent secured object *customer*. So, a policy conflict occurs from the ontology inference. The Access Evaluator will work with the OWL reasoner to detect policy conflict. Then, the Conflicts Manager applies the *Resolution Rule for Policy Conflicts* to resolve the policy conflict. ACP_{65} will be applied to the *preference* object since ACP_{65} is more restrictive than ACP_{55}**.**

CONCLUSIONS

In this paper, I have proposed five requirements for good access-control solutions for enterprise ontologies and a comprehensive framework that meets the five requirements. The proposed access-control properties build access control policies into EO.

In the near future, I will finish the design and implementation of the prototype access-control system. The prototype will be built on Java platform. The OWL2 will be used as ontology language. The SPARQL will be the query engine. The prototype will automatically generate and maintain access-control properties. Also, the prototype will automatically detect and resolve policy conflicts. Additionally, using the prototype, I will provide a thorough evaluation to the proposed solution. Finally, I will extend the innovations to the access control for ontologies in general.

REFERENCES

1. Abel, F., De Coi, J.L., Henze, N., Koesling, A.W., Krause, D. and Olmedilla, D. (2007) Enabling Advanced and Context-Dependent Access Control in RDF Stores**,** *ISWC/ASWC*, 2007, LNCS 4825, 1–14.
2. Baader, F., Knechtel, M. and Pe˜naloza, R. (2009) A generic approach for large-scale ontological reasoning in the presence of access restrictions to the ontology's axioms, *Proceedings of ISWC*, 2009, volume 5823 of LNCS, 49–64.
3. Bertino, E. and Kirkpatrick, M.S. (2011) Location-Based Access Control Systems for Mobile Users–Concepts and Research Directions, *SPRINGL '11*, November 1, 2011. Chicago, IL, USA.
4. Bolchini, C., Curino, C.A., Orsi, G., Quintarelli, E., Rossato, R., Schreiber, F. A. and Tanca, L. (2009) And What Can Context Do for Data? *Communications of the ACM*, 52, 11 (November 2009), 136-140.
5. Calvanese, D., De Giacomo, G., Lenzerini, M. and Rosati, R. (2008) View-based query answering over description logic ontologies, *Proceedings of KR*, 2008.
6. Chen, W. and Stuckenschmidt, H. (2012) Access Control for Ontologies, *www.softplant.de/fileadmin/data/.../AccessControlForOntologies.pdf*
7. Damiani, M. L., Dertino, E., Catania, B. and Perlasca, P. (2007) GEO-RBAC: A Spatially Aware RBAC, *ACM Transactions on Information and System Security*, Vol. 10, No. 1, Article 2, Publication date: February 2007.
8. Defense Information Systems Agency (2004), Database Security Technical Implementation Guide, *Department of Defense*, Retrieved January 31, 2010, from http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

9. Duan, S., Kementsietsidis, A., Le, W. and Wang, M. (2012), Enforcing Query Policies Over Resource Description Framework Data, *US Patent 2012/0047114 A1*, Feb.23, 2012.
10. Fox, S. M., Barbuceanu, M. and Gruninger, M. (1995) An organisation ontology for enterprise modeling: Preliminary concepts for linking structure and behavior, *Computers in Industry*, 29, 1996, 123- 134.
11. Guinn, B. (2011) The Next Generation Consumer Business – Semantically Enabled for Real-Time Intelligence, *CTO Product Enablers*, Amdocs Product Business Unit http://semanticweb.com/sessionPop.cfm?confid=62&proposalid=4133
12. G¨otz, M. (2012) On User Privacy In Personalized Mobile Services, A Dissertation Presented to the Faculty of the Graduate School of Cornell University.
13. Hu, V.C., Ferraiolo, D.F. and Kuhn, D. R. (2006) Interagency Report 7316: Assessment of Access Control Systems, *Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology*, Gaithersburg, MD 20899-8930, September 2006.
14. Jain, A. and Farkas, C. (2006) Secure Resource Description Framework: an Access Control Model, *SACMAT'06*, June 7–9, 2006, Lake Tahoe, California, USA.
15. Knechtel, M. (2008) Access rights and collaborative ontology integration for reuse across security Domains, *Poster*, ESWC 2008 PhD Symposium.
16. Ma, S., He, J. and Gao, F. (2012) An Access Control Model based on Multi-factors Trust, *JOURNAL OF NETWORKS*, VOL. 7, NO. 1, JANUARY 2012.
17. Pavlov, G. (2011) Access Control On Shared Ontologies, *http://www.ibis-journal.net ISSN:1862-6378*.
18. Reddivari, P., Finin, T. and Joshi, A. (2007) Policy-Based Access Control for an RDF Store, *University of Maryland*, Baltimore County, Baltimore MD USA.
19. Uschold, M., King, M., Moralee, S. and Zorgios, Y. (1998) The Enterprise Ontology, *The Knowledge Engineering Review*, Volume 13, Issue 01, March 1998, 31-89.
20. Zhang, G. and Parashar, M. (2003), Context-aware Dynamic Access Control for Pervasive Applications, *Proceedings of the Communication of ACM*, 2004.