Using semantic policies for ad-hoc coalition access control

Anand Dersingh  
Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada, B3H 1W5  
dersingh@cs.dal.ca

Ramiro Liscano  
Faculty of Engineering and Applied Sciences, University of Ontario Institute of Technology, Oshawa, ON, Canada, L1H 7K4  
Ramiro.Liscano@uoit.ca

Allan Jost  
Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada, B3H 1W5  
jost@cs.dal.ca

Abstract— Coalition access control models are required in order to properly manage access to resources among different collaborating organizations. When these relationships are long term inter-organizational agreements and policies can be established that can satisfy appropriate access to the resources owned by those entities. When these coalitions are spontaneous access rights to resources among the parties in the coalition need to be specified by users and must be context dependant. A good example of this is in ad-hoc collaborative scenarios. Controlling access to private services being shared within the collaborative group is a challenge in these scenarios. This paper presents a semantic web approach in order to represent context that can be shared and used by a policy engine to form dynamic groups based on the context of the situation, as well as manage access to the private web services that each group introduces to the ad-hoc collaborative environment. The approach leverages the distributed policy framework (Rein) built on top of a rule-base reasoner (CWM).

Keywords: Access Control, Context Representation, Semantic Web

I. INTRODUCTION

Computing is moving towards pervasive, ubiquitous environments in which devices, software agents, and services are all expected to seamlessly integrate and cooperate in support of human objectives – anticipating needs, negotiating for service, acting on our behalf, and delivering services in an anytime, anywhere fashion [1]. The increase in portable devices creates a situation where user context, such as physical location and activities, is more dynamic. This introduces a new class of services called context-aware services which take user’s context into account. This is very evident in ad-hoc collaborative scenarios where users wish to share ideas and resources at the time that they meet.

In pervasive computing environments, technologies are expected to be invisible from the user point of view. This is why contextual information plays an important role. The idea of using contextual information came from human to human communications. When humans interact with humans, they are able to use implicit situational information, or context, to increase the effectiveness of the conversation. This ability to convey contextual information from human interaction with computers increases the number of new Internet applications and services.

Details in security and network management also need to be hidden from users but have to effectively allow users to gain access to the shared services pervasively without difficulties. This is extremely challenging when entities from different organizations are trying to collaborate. Protecting and managing network services suffer from a set of architectural and integration problems Policy-based approach addresses these problems by providing a means by which the administration process can be simplified and largely automated.

The common goal of both pervasive environments and policy-based systems is the automation process. Moreover, the pervasive environments are based on context, dynamic and unpredictable, and cannot be configured beforehand. This is why policy-based approach is best suite to facilitate the management of entities in the pervasive environments.

An example would be users that are participating in a SIP call that would like to share resources or services among themselves while the call is still in place illustrated in Figure 1. These users in both sites can dynamically form a group consisting only of those participating in the SIP call regardless of their location. The context that has brought these persons and resources together is the SIP call.

Figure 1: Sharing services in inter-organization

A significant challenge is in controlling access to the private services for this particular scenario. Several factors come into play of which the most significant ones are that external access to the private services are defaulted to “no-access” and participants in the call should be able to share their resources under controlled conditions while the call still in place. In other words, as shown in Figure 1, a user in site A may want to share his personal services to the outsiders under the condition that they are participating in a SIP call with the
user in site A. The problems arise due to the fact that a firewall may block the outside access to the printer. This problem can be resolved using rudimentary approach by open up the port accessing the services at the firewall so that outsiders can easily gain access to the printer in site A.

This rudimentary approach may result in security breach to the network as well as it does not satisfy with the goal of setting up dynamic access control based on the situational context such as participating in the SIP call. Policy-based access control is an approach that helps in terms of automating and in configuring devices to allow selective access but the policy must be able to understand and use the context, in this case the “SIP Call”. It is not difficult to develop a proprietary policy-based mechanism to understand and use the context. However, this proprietary mechanism will be tied to a specific application which cannot be reused under several different circumstances. Also, if a new context has to be added at later time, then it is a costly process. This is the reason that a formal representation of the context is needed.

Role-based access control (RBAC) is an approach that is widely used in organizations. A role hierarchy model simplifies the task of managing an enterprise network. A user authorization can be accomplished by first assigning users to existing roles, and second assigning access privileges for objects to roles. Roles also provide least privilege by ensuring that authorized users can access only the resources connected with assigned privileges. However, roles are static information, in other words, roles are the static context which is not suitable to a scenario described above, and there would be no role of the SIP Call. This is why it is important to have context-aware access control in such a dynamic situation.

In this paper, we introduce context-aware access control policy in controlling access based on the context of the situation. To achieve this first domain knowledge must be represented in a formal way so that a policy engine can capture and infer the current situational context. We use semantic representations to formally represent knowledge of the domain as well as inference engine to infer the current contextual information. As well as, we use semantic rule language to form context-aware access policy.

This paper is composed in the following manner: In Section 2, we give background knowledge of semantic web and policy-based network management follows by the proposed approach in Section 3. Section 4 describes the system framework. Section 5 gives information on related work in this research area and Section 6 is a conclusion.

II. BACKGROUND

A. Semantic Web

The Semantic Web is a vision for the future of the Web, in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web [21]. Every item on the Web must be identified by a Uniform Resource Identifier (URI). A URI can be for anything, and anything that has a URI can be said to be on the Web. There is one type of the URIs that is well-known: the URL or Uniform Resource Locator. A URL is an address that leads to a webpage. In other words, it tells a computer where to find a specific resource. Unlike other forms of URIs, a URL both identifies and locates [1].

XML was designed to be a simple way to send documents across the Web. It allows anyone to design their own document format usually by including markup to enhance the meaning of the document’s content. This markup is machine-readable, that is, programs can read and understand it. Basically, using XML-styled documents makes them powerful [19]. XML allows anyone to design their own markup language and it might be the case that others use the same words with different meanings. To prevent confusion, markup elements must be uniquely identified by assigning a URI to each of elements and attributes. This is called XML Namespaces [19].

XML is mainly to make documents for machine-readable but not machine-processable. Machine-processable can be achieved by using Resource Description Framework (RDF). An RDF statement has three parts: a subject, a predicate, and an object. RDF statements can be written in N-Triples or Notation3 [20] which are languages to write simple RDF. However, RDF can be written in XML representation as well, which is a bit more complicated, but says the same thing. RDF is ideally suited for publishing databases to the Web, since most databases contain machine-processable information. Again, when we put them on the Web, we give everything in the database a URI [19].

All the work on databases assumes that the data is nearly perfect. Any system that is hard-coded will likely have limited usefulness, as new terms are invented and defined. In addition, there is no way for a computer or human to figure out what a specific term means, or how it should be used. The use of URIs is useless if we never describe what they mean. This is where schemas and ontologies come in. A schema and an ontology are mechanisms to describe the meaning and relationships of terms. This description helps computer systems use terms more easily, and decide how to convert between them [19]. Originally, RDF Schemata and the DARPA Agent Markup Language with Ontology Inference Layer (DAML+OIL) were developed to solve this problem.

As of [22] an ontology defines the terms used to describe and represent an area of knowledge. Ontologies are used by people, databases, and applications that need to share domain information (a domain is just a specific subject area or area of knowledge, like medicine, tool manufacturing, real estate, automobile repair, financial management, etc.). Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them. They encode knowledge in a domain and also knowledge that spans domains. In this way, they make that knowledge reusable.

The Semantic Web needs ontologies with a significant degree of structure. These need to specify descriptions for the following kinds of concepts [22]:

- Classes (general things) in the domains of interest
- The relationships that can exist among things
- The properties (or attributes) those things may have
Ontologies are usually expressed in a logic-based language, so that detailed, accurate, consistent, sound, and meaningful distinctions can be made among the classes, properties, and relations. Some ontology tools can perform automated reasoning using the ontologies, and thus provide advanced services to intelligent applications such as: conceptual/semantic search and retrieval, software agents, decision support, speech and natural language understanding, knowledge management, intelligent databases, and electronic commerce [22].

Ontologies figure prominently in the emerging Semantic Web as an approach of representing the semantics of documents and enabling the semantics to be used by web applications and intelligent agents. Ontologies can prove very useful for a community as a way of structuring and defining the meaning of the metadata terms that are currently being collected and standardized. Using ontologies, tomorrow's applications can be "intelligent," in the sense that they can more accurately work at the human conceptual level [22].

B. Policy-based network management

Policy-based Network Management (PBNM) has been proposed as a solution to network management problems such as network congestion, administrative challenges, network complexity, security, etc. PBNM is considered as a paradigm in network management by managing network as an entity itself instead of managing the individual components. Figure 1 shows the basis of PBNM [15].

![Figure 2: The basis of Policy-based Network Management [15.]](image)

Principally, a network manager creates a set of policies to define how resources or services in the network can be used. Policies are in the form: if (policy-condition) then (policy-action) [26]. A policy-condition is a script that results in boolean to determine whether an element is a member of a set of elements upon which an action is to be performed. A policy-action is an operation performed on an element or set of elements. These policies are often executed on or near managed devices where the elements live and where operations on those elements will be performed such as configuring a firewall to allow certain application to pass thru.

The two main architectural components for PBNM are Policy Enforcement Point (PEP) and Policy Decision Point (PDP). Figure 2 shows a simple configuration involving these two components [15]. The PDP is a process that makes decisions based on policy rules and the state of the services those policies manage. The PDP is responsible for policy rule interpretation and initiating deployment. It may also be responsible for triggering detection and handling, rule location and applicability analysis, network and resource-specific rule validation and device adaptation functions [15]. In most cases, the PDP transforms the policy decisions and passes to the PEP in a form and syntax that the PEP can accept. However, the PDP may use additional mechanisms and protocols to achieve additional tasks such as retrieving policy information i.e. from policy repository.

![Figure 3: General architecture of PBNM [15.]](image)

PEP is a component at a network node and PDP is a remote entity that may reside at a policy server. The PEP represents the component that always runs on the policy-aware node e.g. device. It is the point that enforces a policy decision and makes configuration changes according to the policy decision. Network nodes can be categorized into groups: policy-aware and policy-unaware. For policy-aware nodes, the PEP is usually a component residing in the nodes. But policy-unaware nodes are nodes that are unable to interpret or form policy information. However, they can still participate in a policy-based network if there is an enforcement mechanism that performs translation from policies to device configuration. As depicted in Figure 2, the translation is performed by a policy-aware agent.

According to the IETF Policy Based Network Management MIB [26], the interaction between the PDP and PEP starts with the PEP that receives a notification or a message or an event that requires a policy decision. The PEP then formulates a request based on the notification received and sends to the PDP. The PDP consults policy repository, makes decision, and returns the policy decision to the PEP. Besides, the PDP may also return additional information that may or may not be associated with the decision.

In some cases, the general architecture of PBNM shown in Figure 3 may not be sufficient as it might be necessary to apply local policies such as an access control list, in addition to the policies applied to the remote PDP. Moreover, it is possible for the PDP to be co-located with the PEP at the same node as depicted in Figure 3b. Both architectures as shown in Figure 2 and 3 illustrate the capability of policy management in a centralized and distributed manner respectively. On one hand, a centralized policy server, which could be responsible for policy decisions on behalf of multiple nodes in an administrative domain, might be implementing policies of a wide scope. On the other hand, policies which depend on local information and conditions might be better implemented locally at the node.

![Figure 4: Local decision point at a device and Co-located PEP and PDP](image)
The language used to specify the policies implemented by the PDP is another vital importance. The number of policies applicable at a network node might potentially be quite large. At the same time, these policies will exhibit high complexity, in terms of number of fields used to arrive at a decision. Various policy languages have been proposed such as Ponder, XACML, Rei, Rein (Rei in N3) [24]. These policies ensure a coherent and consistent application of policies as well as ensure unambiguous mapping of request to a policy action. They also permit the specification of the sequence in which different policy rules should be applied and the priority associated with each one.

III. PROPOSED APPROACH

The purpose of this paper is to adopt the use of semantic web in order to develop a system that performs context-based access control. In other words, context of the situation will be captured to form a group and only members of the group will be allowed access to the services associate with the group.

A. Knowledge Modeling and Representation

Representing domain knowledge could be done from application to application depends on the developers. However, in this way, it would require the knowledge tight to the applications as well as lack of interoperability.

Ontologies are the key to represent context in a formal way due to the following reasons [1]: 1) a common ontology enables knowledge sharing in an open and dynamic distributed systems, 2) semantically well defined ontologies provide a means to reason about contextual information, and 3) explicitly represented ontologies allow interoperable among devices.

Ontologies are critical for applications that want to search across or merge information from diverse communities. Although XML DTDs and XML Schemas are sufficient for exchanging data between parties who have agreed to definitions beforehand, their lack of semantics prevent machines from reliably performing this task given new XML vocabularies. The same term may be used with (sometimes subtle) different meaning in different contexts, and different terms may be used for items that have the same meaning. RDF and RDF Schema begin to approach this problem by allowing simple semantics to be associated with identifiers. With RDF Schema, one can define classes (concepts) that may have multiple subclasses and super classes, and can define properties, which may have sub properties, domains, and ranges. In this sense, RDF Schema is a simple ontology language. However, in order to achieve interoperability between numerous, autonomously developed and managed schemas, richer semantics are needed. For example, RDF Schema cannot specify that the Person and Car classes are disjoint, or that a string quartet has exactly four musicians as members [22].

Web Ontology Language (OWL) is intended and designed to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called an ontology. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to represent machine interpretable content on the Web. OWL is a revision of the DAML+OIL web ontology language incorporating lessons learned from the design and application of DAML+OIL [21].

Each domain is a knowledge model about facts in which entities in the physical world such as users, devices, and services interact. The knowledge model of domain consists of the contextual information that describes at least users, devices, services and activities that users are engaging. The modeling of the users, devices and other entities in the domain are straightforward in the sense that they can be described by using one of the semantic languages. However, for web services which are normally defined by Web Service Description Language (WSDL) which is an XML based document cannot be inferred or processed by the machine.

In order to utilize the use of WSDL in the semantic web area, it is clear that a WSDL RDF mapping approach [25] can be brought to the system which in turn better than reinventing the wheel by modeling the structure of the service description.

B. Context-based Access Policy

Policies are usually written in the form of rules. Each policy must be associated with the domain knowledge. In a pervasive environment, a policy must be able to use the context of the situation in order to perform access control. Therefore, it is necessary to reason over the domain knowledge to obtain the context.

Inferring knowledge from the knowledge base or facts is the crucial part of the work in this area. Rule-based languages have played a dominant role in reasoning on the Semantic Web. In this paper, we introduces context-based access policy which is an access control policy but written in a form of rules to be able to capture the situational context.

Context-based access policies are a set of rules which describe access rights. The access policies may or may not require context information to allow access but can support contextual data. If the policy requires context, it will be written with a set of rules which identify which contexts are under consideration. In other words, these rules provide the parameters to reason over the KB in order to obtain the accurate information of the context. For example, a context rule would be if any user has the same Call-ID in a SIP Call, then they are in the same SIP Call. The keyword “in the same SIP Call” is the context of the situation

IV. SYSTEM DESIGN AND IMPLEMENTATION

The approach taken in this paper is to use semantic web in order to represent domain knowledge as well as a rule-based language in reasoning on the Semantic Web such as N3. Rein which is based on N3 was proposed [24] as a generic framework for representing rule-based policies.
Rein is a decentralized framework for representing and reasoning over distributed policies in the Semantic Web. Rein uses high level Rei [18] concepts for policies and N3 rules to connect these policies to each other and the Web. Policies in Rein use information defined in and inferences made by other policies and web resources forming interconnected policy networks. Rein allows policies to be represented in different policy languages and uses N3 rules, a semantic web rule language, for defining the connections in these networks. Reasoning over these networks to obtain policy decisions is done using CWM and N3 reasoner. Rein consists of three main components – a high level ontology for describing policy networks, mechanisms in N3 for using information and inferences from both policies and web resources, and an engine for inferring policy decisions from policy networks.

In this paper, we adopt Rein as a platform as a proof of concept. This is due to the fact that Rein allows knowledge or ontologies to be described RDF-S, RDF/XML, OWL, or N3 which corresponds to our approach in order to use semantic web as a way to represent knowledge. In addition, Rein is able to interconnect policies to form policy networks; we adopt this mechanism in order to connect access policy and context policy together in order to provide context-aware policy system.

A. System Architecture

Figure 5 shows the system architecture. The following describes each component of the system:

1) Knowledge Base

Knowledge Base (KB) is a data repository of domain ontology. An ontology defines a common vocabulary to share information in a domain. It includes definitions of concepts and relationships among them [13]. In this work, it is necessary to build ontologies of the domain. We use semantic web languages like RDF, RDFS, and OWL to model entities in the network domain. These semantic markup languages are for defining domain ontologies. An ontology is a set of definitions of classes, concepts, and properties, relationships of the entities in the domain. The domain of interest here is a network domain such as shown in Figure 1. The entities in the domain include users, devices, and services. However, for the services, we use WSDL RDF mapping [25]. WSDL RDF mapping provides a set of markup language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-interpretable form. In other words, it provides a mechanism in describing an ontology of services that includes service descriptions.

2) Knowledge Handler

Knowledge Handler (KH) is responsible for acquiring and monitoring events in the real world that may affect the facts in the KB database. With a dynamic characteristic of pervasive environments, a change of the situation has an effect on the usability of the system. The primary task of the Knowledge Handler is to perform assertion to the KB database according to the model of the domain knowledge as well as maintain the consistency of the KB since the KB has to accurately reflect the dynamic changes of the environment.

Another role of the Knowledge Handler is to notify the Context-Aware Policy Decision Point of the changes of the environment. This is because context-aware policy decision point uses the information in the KB to make decision. The changes in the real world situation may affect the decisions that have been made. For example, if the policy allows those who are participating in a conference to share the services among themselves if one hang up that person must not be able to gain access to the services shared in the conference called.

3) Context-Aware Policy Decision Point

Context-Aware Policy Decision Point is a policy decision point that can capture the context and make a decision based on the set of policies which may require context of the situation. There are two parts that form the Context-Aware Policy Decision Point which are Context-based Access Policy and Inference Engine.

Context-based access policy describes access rights based on the rules given in the policy. This access policy may or may not require context to allow access depends on the resources’ owners. If the access policy requires context it must be semantically written, using rule language, in order to inform the inference engine to infer the situational context. In other words, access policy provides the rules to reason over the KB in order to obtain accurate information. There are several query languages that can be used to query or infer the information in the KB but here we use N3 rule language [24] to form access policy as well as a rule itself can be used to query information from KB.

Inference engine is the part that tries to derive answers from the KB by first interpreting the rules given in access policy. In this paper we use Common World Machine (CWM) which is a forward chaining inference engine [24].

Figure 5: System Architecture

B. Implementation

The current implementation of this work is focusing on the modeling and representation of the domain knowledge as well as context-based access policy in order to control access based on the context of the situation. We use the scenario as shown in Figure 1 that a user in site A allows a user in site B to gain access to the web services as long as there is a SIP Call going...
on between these two users. This work does not restrict to this specific scenario but we use it as a proof of concept in order to justify our work.

Figure 6 shows partial ontologies of users, device, services as well as relationships among them. Starting with services will be on devices such as server. There are also many kinds of services such as phone services, print services, or web services. A person may own a service as well as may be in a SIP Call.

![Figure 6: Partial graphical knowledge representation](image)

The following shows domain ontology based on graphical representation in Figure 6 written in N3.

```n3
@prefix : <#>.
:Person a rdfs:Class.
:Device a rdfs:Class.
:Service a rdfs:Class.
:Printer a rdfs:Class; rdfs:subClassOf :Device.
:PrintService a rdfs:Class; rdfs:subClassOf :Service.
:PhoneService a rdfs:Class; rdfs:subClassOf :Service.
:SIPCall a rdfs:Class; rdfs:subClassOf :Service.
:CallID a rdfs:Class.
:incall a rdfs:Property;
  rdfs:domain :PhoneService.
:owns a rdfs:Property;
  rdfs:domain :Person;
  rdfs:range :Service.
:hasSIPURI a rdfs:Property;
  rdfs:domain :Person; :Service;
```

For web services we use WSDL RDF mapping [25] which is written in RDF and OWL. Even though the ontology is not written using N3 but in Rein framework is able to combine information in different format. The following shows partial web service representation written in N3.

```n3
@prefix rwsdl: <http://www.w3.org/2005/10/wsdl-rdf#> .
@prefix : <#>.
:projectService a rwsdl:Service ;
  rwsdl:endpoint projectEndpoint.
projectEndpoint a rwsdl:Endpoint ;
  rwsdl:address <http://corona.cs.dal.ca/project> .
```

C. Policy ontologies

Policies are a means to dynamically regulate the behavior of the system components without changing code and without requiring the consent or cooperation of the components being governed. By changing policies, a system can be continuously adjusted to accommodate variations in externally imposed constraints and environmental conditions. The adoption of a policy-based approach for controlling a system requires an appropriate policy representation and the design and development of a policy management framework.

```n3
{ ?OWNER a ont:Person.
  ?SIPCALL a ont:SIPCall.
  ?OWNER ont:incall ?SIPCALL.
  ?OWNER ont:owns ?DEVICE.
  ?DEVICE a ont:Device.
  ?SERVICE ont:target ?DEVICE.
  ?WHO a ont:Person.
  ?WHO ont:incall ?SIPCALL.
} => {?WHO reina:ispermitted ?SERVICE}.
```

V. RELATED WORK

The typical access control models: DAC, MAC, and RBAC have been explained in Section 2.1. These models were developed, mainly, for closed and static environments where context of the situation is fixed. It is not feasible to apply these models in collaborative and dynamic environments that require knowledge of the situation.
Thomas proposes the notion of Team-based Access Control (TMAC) [9] as an approach of applying role-based access control in a collaborative environment. He takes a notion of “team” as an abstraction that encapsulates a collection of users in specific roles with the goal of accomplishing a specific task. The team components have a set of users in various roles. He defines “Team role” which is a set of roles defined within a team to restrict the role memberships and “Team permission” which is a set of permissions that are defined across team roles and objects. In addition, a context of a team contains users context and object context.

Wang [10] argued that the TMAC model defined by Thomas is not clear and does not explain how it incorporates the team concept into a general RBAC framework, as well as, he argues that there is no need for team roles and the team could be set up on project basis. Wang presents the integration of RBAC in a team-based organization context. Team is defined as a group of users working together.

The TMAC models proposed by both Thomas and Wang address the problems of access control in collaborative environments. The environments under consideration are still a closed environment. TMAC also adopts the concept of RBAC which is used mainly in corporate domains. It does not address the issues of dynamic environments.

Yagüe et al. [8] has presented the application of Semantic Web concepts to the access control area. They use the Semantic Access Control Model (SAC) as a model to reach interoperability through the semantic integration in heterogeneous and distributed environments. The SAC uses different layers of metadata to take advantage of the semantics of the different components relevant to access decisions. The model is based on the use of semantic descriptions of the authorization entities; separation of the attribute certification and the authorization management functions, following the layers infrastructure of the Semantic Web. The SAC was implemented based on Semantic Policy Language (SPL) by describing, semantically, access control criteria, allocation of policies to resources and semantic information i.e. properties about resources and context. Along with the implementation is autonomous enforcement mechanisms called XSCD (XML-based Secure Content Distribution) was developed.

Xiaopeng et al. [6] has also proposed a semantic access control approach in Grid computing in order to authorize and administrate Grid access requests. Generally, Grid refers to systems and applications that integrate and manage resources and services distributed across multiple control domains. In other words, the Grid should provide access for users and systems from various places or domains. They use semantic representation to describe policies, requests, resources, and other entities, as well as use machine reasoning at a semantic level to determine whether the requests should be passed.

The semantic access control approaches in [6], [8] address the problem of interoperability in open and distributed environments. Even though they use the concept of Semantic Web in terms of representing entities in the systems, they still do not address an access control in the context-aware environments as well as the issues of the policy management.

There has been little work done on context-aware access control. The certificate approaches have been proposed with the use and proliferation of attribute certificates and there is no standard for attribute certificates [14]. Moreover, the use of PKI requires authority or a set of authorities to manage certificates which does not scale well in the Internet. In terms of intranet and extranet, it is feasible to use distributed certificate management [14] but still lack of context-aware.

R. Bhatti et al. [17] has proposed an extension to XML-based Generalized Temporal Role Based Access Control (X-GTRBC) which is based on X-RBAC, in order to support context-aware access control for Web services environments. X-GTRBC was originally proposed as a solution to enterprise-wide access control. In addition, they proposed trust-based role assignment to users. The framework relied on the Trust Management (TM) approach of trusted third parties such as Public Key Encryption Certificate Authority (PKI CA), and uses the certification provided by them to assign roles to users. However, the approach relied on trust between domains and introduced a set of algorithms to evaluate service access request (make decision). Our approach uses the policy-based approach for making decision which is suit if there are multiple policies apply to a resource.

G. Sampemane et al. [7] presented an access control architecture for Active Spaces where Active Spaces are interactive environments such offices or meeting rooms. They introduced the notion of collaborative access control modes and developed appropriate access control policies that augment traditional modes of collaboration. An Active Space may be in an individual, group supervised, or collaborative mode depending on the users, devices, and other context in the space. The system was relied on the RBAC model for policy configuration and implementation. However, this work does not address directly to the context of the environment such as user’s activities as well as how access control policies can be derived from high level policies.

One of the research work in the area of context-based access control is Ubiquitous Context-based Security Middleware (UbiCOSM) [27] that adopts context as the principal for security policy specification and enforcement processes. Unlike traditional access control model which is subject-based, access control permissions are directly associated with contexts. UbiCOSM adopts an RDF-based format for context representation to over heterogeneity of data representation. However, it does not adopt RDF-based format as a means to infer the relationships of entities. In other words, it does not support the extraction of the context from the primary context.

VI. CONCLUSION

This paper proposes context-aware access control for ad-hoc coalition access control by using semantic web. We model and represent domain knowledge by using semantic web languages such as RDF, RDFS, and OWL. We also form context-based access policy by using semantic rule language in order to infer the current context of the situation or activities of the entities in the domain. This access policy is built on top of Rein policy framework.
We address the issues of controlling access in a dynamic environment such as ad-hoc collaboration where the context used is dynamic such as current user’s activities. This sort of problem cannot be resolved with existing approach such as RBAC since the contexts used for RBAC are roles and RBAC itself cannot capture and use the current situational contexts in the access policies.

Currently we have designed and implemented the domain knowledge as well as context-aware policy decision point. As a result we can now make decision based on the context of the situation such as participating in a SIP Call. However, it is not restricted to this particular scenario. We can also model and capture others activities to use as a context in access policies.

REFERENCES


