SECURITY AND AUTHORIZATION ISSUES IN HL7 ELECTRONIC HEALTH RECORDS: A SEMANTIC WEB SERVICES BASED APPROACH

by

RICHARD SCOTT PATTERSON

(Under the Direction of Dr. John A. Miller)

ABSTRACT

Semantic Web services are the next generation of Web services, offering the possibility of an automated Service Oriented Architecture. Much work has been done in the areas of Semantic discover of services based on the semantics of t he operations available through services. There has been further research in this area using the inherent semantics associated with Quality of Service in an effort to discover and choose the appropriate Semantic Web service for a given situation. Because Web services in general expose an organizations infrastructure via this Web interface, there has been a great deal of effort placed in specifying Web service security. Three fundamental standards are the WS-Security, WS-Encryption, and WS-Signature specifications. The work herein focuses on authorization in a Semantic Web services environment. We will look at discovering an appropriate service in which we predict authorization through the use of semantic annotations.

INDEX WORDS: Web Services Discovery, Authorization, Semantic Matching of Web Services, Ontology-based matching of Authorization, Web Service Authorization Discovery, Semantic Web Services

SECURITY AND AUTHORIZATION ISSUES IN HL7 ELECTRONIC HEALTH RECORDS: A SEMANTIC WEB SERVICES BASED APPROACH

by

RICHARD SCOTT PATTERSON

Bachelors in Business Administration, University of Georgia, 1998

A Thesis Submitted to the Graduate Faculty of the University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2006

SECURITY AND AUTHORIZATION ISSUES IN HL7 ELECTRONIC HEALTH RECORDS: A SEMANTIC WEB SERVICES BASED APPROACH

by

RICHARD SCOTT PATTERSON

DEDICATION

I would like to dedicate my thesis work to my mother who gave me great support and encouragement through this process; and to Kevin and Lynny Corcoran for their personal support and guidance.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank Dr. Miller whom is my Major Professor and co-author of the journal submission. He has been helpful through this process and understanding of the difficulties I have had recently in my personal life. I would also like to thank Dr. Sheth for his support and interest in this project. In addition I would like to thank Dr. Arpinar for joining my committee.

Thanks to J. Mike Davis and Ed Coyne from VHA and INCSTI, respectively, for their involvement; along with Rick Kuhn from NIST.

TABLE OF CONTENTS

Chapter 1

INTRODCUTION AND LITERATURE REVIEW

Web Services and the Service Oriented Architecture are impacting the ways in which businesses interact with the world. Semantic Web services have the potential to substantially impact businesses and individuals. Security may be the final last challenge to Semantic Web services. An important aspect of security in Semantic Web services is authorization which icurrently a research area of great interest, even while the WS-Authorization specification remains to be standardized [IBM, Microsoft 2002].

Authorization is determining who has permission to use which resources. Web services expose critical data and organizational infrastructure via their interface on the Web. It is obviously not wise to permit anyone and everyone to use sensitive Web services. In order for Semantic Discovery of Web services [Dogac et al., 2002] [Verma, 2005] to be appropriately discovered, there needs to be a framework which will indicate authorization for a service.

[Yague et al., 2003] has implemented an access control structure for Semantic Web services, while [López et al., 2005] has a similar approach for distributed heterogeneous networks using Extensible Access Control Markup Language. An approach which is similar to the one presented here is [Kagal et al. 2004]. The authors use a concept ontology and a rule based ontology to describe access control policies.

The work presented here focuses on Semantic description of authorization. Semantic annotations are added to WS-Policy in order to describe authorization constraints for a Web service. This is done in a similar way as is done in WSDL-S [Akkiraju et al., 2005] ans SAWSDL [SAWSDL, 2006]. These annotations aid in the Semantic discovery of Web services through Semantic matching and constraint analysis, effectively allowing potential requesters to determine if they will have authorization to use a service.

Chapter 2

SECURITY AND AUTHORIZATION ISSUES IN HL7 ELECTRONIC HEALTH RECORDS: A SEMANTIC WEB SERVICES BASED APPROACH

[1] Richard S. Patterson, John A. Miller, Jorge Cardoso, and Mike Davis. To be submitted to International Journal of Web Services Research

ABSTRACT:

As Semantic Web services begin to emerge as the next evolution of the Service Oriented Architecture, it is become clear that authorization is going to be one of its biggest challenges. Not only are there the typical obstacles which most areas of Semantic Web services have had to overcome, i.e. what parts of a Web services need semantic information, how best to use the semantics, and agreeing on standards, but there are the fine grained security implications as well. For instance, how much authorization information is necessary to aid in Semantic Discovery of Web services? Is the authorization information opening any new security holes? It is the goal of this article to address these issues by providing a framework for expressing the proper authorization information in order to aid in the Semantic Discovery of Web services in which the requesting service most likely has the authority to invoke.

KEY WORDS:

Web Services Discovery, Authorization, Semantic Matching of Web Services, Ontology-based matching of Authorization, Web Service Authorization Discovery, Semantic Web Services

2.1 Introduction

2.1.1 Web Services

Web services are being integrated into many organizations as part of their Service Oriented Architect (SOA) which includes application integration, business-to-business processes, and e-business solutions. Web services are a loosely coupled technology, allowing cross platform interoperability [Shivaram, 2003]. Web services can be linked together to form a complex service, otherwise know as a process [Leymann, 2002]. Web services are discoverable [UDDI Spec Technical Committee Specification, 2002] and can be easily assembled into a process and then disassembled once the process execution has ended. The discovery of Web services is made possible through the XML description of a service. The standard for the XML description of a service is the Web Service Description Language or WSDL [Christensen et al., 2001]. Invocation of a Web service is the invocation of an operation which has been made available.

Web services can be used to expose inter-organizational components such as business critical data, business processes and internal workflows [Shivaram, 2003]. Organizations may expose some of these components in order to capitalize on the cost savings and reduced complexity that Web services can add to there SOA. Because the SOA is more dynamic, loosely defined, and ubiquitous, new security measures are needed to protect key business information. There are currently standards proposed or accepted regarding authentication, encryption, and identity management. These areas of Web service security use a combination of tried-and-true technologies such as keys, username token, and RSA encryption, along with newer technologies such as XML signature [XML-Signature, 2002] and SAML (Security Assertion Markup Language) [SAML 2.0, 2005]. In Section 3, these technologies will be reviewed regarding how they are applied in Web services. The focus of this paper is authorization in Web services. Specifically, we address authorization in Semantic Web services discovery [Verma, 2005]. We will demonstrate how to express authorization constraints semantically and use this information in the discovery process.

Web services are a dynamic environment and Semantic Web services goal is to describe the functional and non-functional requirements of Web services. Determining whether a potential client has authorization to invoke a Web service can be a complex and tedious task for a human to accomplish through manual matching. Obviously this has two shortfalls from the dynamic SOA goal. First, it is error prone and second it is not dynamic. Semantic information is machine interpretable, therefore allowing us to use algorithms to dynamically determine relationships and context. Authorization conditions can be complex and traditional WS-Policy lacks the expressiveness to describe the conditions of access control. Errors can be reduced through this process as well. Since WS-Policy is written in XML, it is difficult for a human read. However, XML is easily parsed by a computer. Using a machine to parse the document and retrieve the semantic information is much more efficient and less error prone than a human.

2.1.2 Why Use Semantics

The WS-Policy specification is a model and syntax for describing the policies of a Web services. It relies on its follow-on specifications, such as WS-Trust, WS-Agreement, WS-Security, and WS-Utility, which make within WS-Policy. The assertions are based on an XML based domain vocabulary. A client and service provider can make assertions in WS-Policy from any domain using the specifications which describe the vocabulary. When matching policies, if the policy matching mechanism is unaware of the domain context then it would be limited to using syntactical matching. Consider the following example where a client and a service provider have included authorization assertions from the Health Care domain.

Service Provider:

- Must be a *Physician* working in *Emergency Service* of the *Health Services* Industry

Client:

- Is an *Emergency Room Physician* working at a *Hospital* in the *Emergency Room*

These assertions are equivalent. The domain knowledge needed to determine that these assertions are equivalent are absent in a purely syntactic matching mechanism. Therefore, using a string matching algorithm would result in the denial of authorization for the client, which is a false negative result. These assertions can easily be determined to be equivalent by using domain information along with semantic reasoning. From our example, it can be determined that $\forall$*Physician WorkIn Emergency Room (Physician)* $\Rightarrow$ *provides Emergency Services*. A matching mechanism which uses domain knowledge captured in an ontology can improve the results over traditional purely syntactic matching mechanisms.

2.1.3 The Current State of Web Service Authorization

Authorization in Semantic Web services and other ubiquitous computing environments is currently a hot research area. Authorization is not involved with validating the identity of a potential client; the term client is used because Web services are an extension of the client-server model. The validity of a client's identification is determined during authentication. During authentication a clients credentials are resolved and the contents of the message are verified. The technologies used in authentication will be briefly discussed in Section 3.

Authorization is verifying that the identity has the proper permissions to access the requested resource. This can be done through querying an access control policy. There are several robust systems used for access control, such as Lightweight Directory Access Protocol (LDAP) and Role Based Access Control (RBAC). Access control is the enforcement of authorization. Once authorization has been determined, an access control system is used to enforce the policies.

Currently, in Web services, including Semantic Web services, the primary means of authorization is through authentication techniques. There are many Web services available which use Username Token pairs to make authorization decisions. This process is simple and similar to signing into a system. The prerequisite is that an account must have been created. There are technologies which allow two parties, previously unknown to each to each other, to Authenticate. It is here that there are two fundamental authorization questions. On the server side, how much information should be shared with an entity to which there is no previous relationship. From the client's perspective, how does the client know if they will have access to the information and resources they seek?

As Web services continue to evolve into Semantic Web services for automated discovery and execution of business processes [Verma, 2005], these two questions become more prevalent. Currently, in Semantic Web services, discovery is based on a semantic description of the Web services' operations and the semantics of the parameters [Akkiraju, 2005]. Therefore, once a Web service has been discovered semantically, how can one assume that authorization is to be 'permitted'? Currently, in Semantic Web services it cannot. However, the approach outlined and detailed in this paper is a system for describing authorization in a Semantic Web services environment. This will add authorization discovery, what we call prediction, to the current Semantic Web services discovery framework.

We call our approach prediction and do not claim to 'know' if a client will have authorization because, as we will discuss in section 3.5.2, there is a variable of uncertainty between what a Web service provider may describe and what a client may assume. Our goal is to limit this variable as much as possible, which will increase the accuracy of our prediction.

2.1.4 Scenario Example

Throughout this paper there will be a running example to show the usefulness of the Semantic Authorization Discovery system. The scenario used in the example is taken from the HL7 RBAC Healthcare Scenarios v 1.0. The scenario is SRD-001, Physician with Review Documentation Privileges [HL7 Security Technical Committee, 2005]. In this scenario, a patient is being seen by a physician for a diabetic consultation. The physician needs access to the patients' medical history and results from tests which are being performed during the visit. As well, any follow up information in the future regarding this visit should be made available to the physician. This scenario will be detailed in Section 2.

In the HL7 scenario, no detail is given about the underlying computer systems. Therefore, in order to make this scenario more applicable to Semantic Web services while keeping relevant to HL7, the computer systems will be discussed here. The general idea is that each department mentioned in the scenario has it own system implementation. This is not unordinary in large hospitals and is even more common in hospitals with multiple locations.

In order for the departments' computer systems to interact with one another, Web services have been implemented. In an effort to enhance the data integration between the departments and between the physicians and patients, semantics have been applied to information stored in those departments and to the Web services which are provided. Furthermore, semantics regarding authorization have been added.

2.1.5 Overview of Authorization Prediction

The discovery of Web services uses a description of the service and key words on metadata to query a UDDI registry with semantic matching capabilities. Semantics add to the discovery process by providing information on the operations, preconditions, and effects of the services. This enables clients to find and invoke the appropriate services more efficiently. However, Semantic Web services still do not provide a means to predict whether the client will be granted permission to access the resource.

The Authorization Prediction analysis is built into the METEOR-S system which is a Semantic Web services and process system [Verma et al., 2004]. Once services have been discovered through UDDI, semantic constraint analysis is preformed on authorization information. This requires that a Web service express its authorization criteria semantically and the semantics from which the criterion is expressed is available to the client (i.e., ontology). Using a domain specific ontology and an authorization specific ontology, a client service can make a prediction on whether they will be authorized to invoke the service which ultimately accesses the resources.

2.1.6 Benefits and Contributions

The first and primary benefit that Semantic Authorization Prediction provides is an efficient means for a client to discover whether it may have permissions to access resources available through Semantic Web services. In a non semantic environment human users must decide this by reviewing the documentation regarding the Web services which they consider invoking. Without proper semantic authorization descriptions, this process can not be implemented automatically in a Semantic Web services environment.

We have created extensible elements in order to add these descriptions as semantic annotations with in either the WSDL or WSP (Web Services Policy) file. Annotations are allowed in the WSDL file to follow the specification for WS-Policy attachment. The novelty of our approach is that these annotations are made through the use of an RBAC ontology.

A benefit is that this approach makes authorization part of the Web services system and not part of the application or resource being accessed. This is important because authorization can be checked prior to the service accessing the contents of the SOAP body. It is less likely that malicious contents will be sent from an authorized user. However, this does not stop malicious contents from being contained in the SOAP header.

Another benefit of our approach is the addition of client authorization information to a BPEL (Business Process Execution Language) WSDL process policy. This is a novel approach to authorization in Semantic Web services. The authorization information is a description of the user, client, who is requesting access to a resource. Because it is a part of the process policy file, the client information can be dynamically injected into a template processes for each client using the process, which provides reusability.

The idea of creating a RBAC ontology is a significant benefit of our approach. This provides an agreed upon standard to describe authorization. Semantic Authorization Prediction incorporates RBAC and XACML [XACML 2.0, 2005] standards. It is because of their acceptance and uses in the authorization domain that make this approach a logical step for Semantic Web services. Other approaches, which we will cover in Section 7, have created their own ways to express authorization; thus recreating the wheel. By using accepted standards, it is easier for access control personnel to use and is more likely to be accepted.

2.1.6 Sections of the Paper

The rest of the paper is laid out as follows. Section 2 discusses a Health Care Application. This begins with a discussion on Health Level 7 in section 2.1. Section 2.2 further explains in detail the scenario which we will be referring to in later sections. Section 3 is a review of Web services security; our goal is to demonstrate that authorization is the last big hurdle. Section 3 has five sections which are: Message Level Protection, Message Privacy, Message Validity, Authentication, and Authorization. Section 4 discusses BPEL which is used to create our Web service process or workflow. In section 5 we examine the HL7 RBAC ontology. Section 6 discusses our implementation. This begins with our architecture. 6.3 explains the Server side while 6.4 explains the Client Side Authorization Prediction. In 6.5, we discuss the evaluation of our techniques. Related work is visited in section 7. Lastly, Section 8 discusses our conclusions and future work in the area.

2.2 Health Care Application

2.2.1 Overview of Health Level 7

Health Level 7 (HL7) is an American National Standards Institute (ANSI) - accredited Standards Developing Organization (SDO). HL7's domain is clinical and administrative data in the healthcare arena. HL7 has members such as providers, vendors, government groups and others. These members are known as the Working Group [HL7]. They have an interest in the development and advancement of clinical and administrative standards for healthcare.

The standards, or specifications, enable healthcare applications to exchange key sets of clinical and administrative data. HL7 is an international community of healthcare experts and information scientists working together to create standards for the exchange, management and integration of electronic healthcare information.

The "Level 7" refers to the top level of the ISO (Open Systems Interconnection) model, which is the application layer. The application layer is concerned with applications, end user processes, quality of service, authentication, privacy, constraints, and the semantics of information. The last four in this list are clearly related to our goals in this paper. This is just one reason that we have chosen to use HL7 in our example.

The HL7 Security Technical Committee has formalized a set of permissions for Role Based Access Control (RBAC). Each permission in the permissions set is a set containing at least one operation and only one object. The permissions set were created using a scenario-based role engineering process. HL7's engineering and role definition content models are compliant with the ANSI RBAC standard. The ground work laid out by HL7 Security Technical Committee enables us to use real world examples in the testing of our system.

In addition, over the past several years the HL7 Technical Committee has been working on standards associated with the exchange of information via Web services. Some of the issues which have kept them from formalizing a standard have been reliability, continually evolving technical standards, and security. We believe that the work presented here will help the latter.

2.2.2 Scenario

In the scenario that follows, the text that appears in caption (i.e., <<Pyy-nnn abstract permission name>>) indicates a reference to the abstract permission name and its unique identifier as described in the tables of the HL7 RBAC Healthcare Permission Catalog. In order to make this scenario more applicable to Web services we will assume that patient medical records are updated and accessed through a Electronic Health Record (EHR) Archive which has been implemented using Web services. To further extend this example to the "real world" we assume that there are several EHR Archives, each containing similar data. This is equivalent to consumer credit reporting. The discovery of the EHR which the physician will use is dependant on the permissions the physician has with each EHR as it pertains to this particular patient. This is practical since not every physician is the primary physician. Physicians who are specialist further demonstrate that granular permissions to a patient's medical record are a reasonable assumption. Therefore, when discovering the appropriate Web service to invoke, the physicians' authorization to access the medical record will be analyzed.

"Mr. Patient was placed in a clinic examination room for a Diabetic Consultation. The Physician greeted Mr. Patient and accessed Mr. Patient's medical records <<PRD-006 Patient Identification and Lookup>> from an EHR. After briefly reviewing Mr. Patient's recent pertinent medical history <<PRD-003 Review Medical History>>, problem lists <<PRD-016 Review Problem Lists>>, health status data <<PRD-014 Health Status Data>>, existing order(s) <<PRD-004 Review Existing Order(s)>>, medications <<PRD-010 Review Patient Medications>>, allergies <<PRD-011 Review Patient Allergies or Adverse Reactions>>, test results/reports (evaluating high and low values) <<PRD-001 Review Patient Testing Reports>>, immunizations <<PRD-013 Review Immunizations>>, and visits <<PRD-012 Review Past Visits>> within the EHR, the

Physician next reviewed Mr. Patient's vital signs, patient measurements <<PRD-005 Review Vital Signs/Patient Measurements>> (e.g., height and weight), and the chief complaint(s) <<PRD-002 Review Chief Complaint>> that were entered for this encounter." [HL7 Security Technical Committee, 2005]

We will use three aspects of this scenario to compose a Web services process for our evaluation and testing. The three aspects we will focus on are 'Patient Identification and Lookup', 'Review Medical History', and 'Review Problem Lists'. All the accesses in the scenario that the physician makes to the patients record can be implemented. However, implementing the entire scenario would not add to the validity of our system. In our implementation it is assumed that 'Patient Identification and Lookup' must be preformed at least once prior to either of the other operations.

As a running example we now describe a detailed 'instance of' use case. We have created three companies to act as Electronic Health Record repositories. Each repository has a Web service implemented for each of the operations from the scenario. The *Client* is an 'Emergency Room Physician and works for a St. Francis Hospital. The services of Company1 had been annotated such that authorization is to be granted to a client that has the *role* 'Physician' working in 'Health Services', *subjectCategory*. The services of Company2 had been annotated such that authorization is to be granted to a client that has the *role* 'Nurse' working in 'Health Services'. The services of Company3 had authorization information which was a combination of the previous companies in which one service grants and two deny authorization. As we will show in section 5, St. Francis is a 'Hospital' is an instance of an 'Organization' involved in 'Health Services'. The details of the process will be further discussed in Section 6.4, Evaluation of Techniques.

2.3 Web Services Security

In securing Web services, there are five fundamental areas to consider; Message Level Protection, Message Privacy, Parameter Checking, Authentication, and Authorization. When examining these areas it is important to stay within the context of Web services and not network security in general. This is because network security is at a different layer of the ISO model; Web service security is at the application layer. As we discuss these areas of security, observe the following. Some of solutions use the same or similar technologies to achieve vigilance. Not all of the technologies used were developed for Web services and may have been around for many years. Which of these areas could benefit from semantics? Four of the five areas have been addressed; however, authorization has not and is currently a hot topic in both Web services, Semantic Web services, and the Semantic Web. In this section, our discussion of Web services Security will be in the context of a single service. In the next section, we will further the discussion to Web service processes.

2.3.1 Message Privacy

Message privacy deals with the confidentiality of a message. Here unauthorized entities should not be able to access the information within the message. It is important to remember here that a part of this information is the XML Signature and Token, found in the message header, which can be seen in Figure 2. To ensure confidentiality an encryption scheme must be implemented.

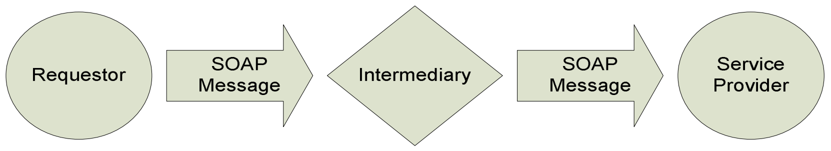


Figure 1 - Soap Message in transit

Since Web services can be chained together to form complex services, traditional point-to-point encryption schemes, such as SSL, do not suffice. Point-to-point schemes work at the Network layer of the ISO model. Therefore, once the message has been received by an entity it is decrypted in its entirety. This entity may be an intermediary and not the provider of the service, Figure 1. Furthermore, a message may cross multiple trust domains due to routing caused by elaborate messaging communications [Web Services Architecture, 2004]. What is needed is an end-to-end encryption scheme.

The XML Encryption standard provides the necessary framework for accomplishing this task. XML Encryption allows for the encryption of any combination of the message body, header, attachments, and sub-structures [XML-Encryption, 2002].

When a message or part of a message is encrypted, the encryption information can be made available in the message header. This information is useful for complex services since each Web service in the chain will need to know how to decrypt the section of the message relevant to their service. This information should not be the actual key to decrypt the message.

An example will clarify. When a requester encrypts a message body and XML Signature information in the header, it may then specify in the header that it has used the providing service's public key. A public key allows for the encryption of data but only the private key may decrypt the data. Once the provider receives the message it sees that the message has been encrypted using its public key. The provider then decrypts the message using its private key.

XML Encryption allows multiple different keys to be used with in a message to encrypt different sections, elements, of the message. Each encrypted section is referenced in the message header and mapped to the key information if provided. This provides end-to-end encryption through intermediaries which may also be accessing the parts of the message.

Table 1 - XML Algorithms

Purpose	**Algorithm**	**Specified as**
Digest	SHA1	Required
Digest	SHA256	Recommended
Signature	DSAwithSHA1 (DSS)	Required
Signature	RSAwithSHA1	Recommended
Canonicalization	Canonical XML (omits comments)	Required
Canonicalization	XML with Comments	Recommended
Transform	XPath	Recommended
Transform	Enveloped Signature	Required

Table 1 above provides an overview of the algorithms specified in the XML-Signature and XML-Encryption standards. Required algorithms are the minimal to comply with the standard; while recommended is just that, recommended.

2.3.2 Message Level Protection

Message Level Protection has to do with message integrity. This means being able to detect when a SOAP message (message) has been modified from its original state and the ability to guarantee that the contents have not been modified [Web Services Architecture, 2004]. This is done by creating a message digest.

A message digest is an encoded message, a cryptographic checksum of an octet stream [WS-Security, 2002], which is created using an algorithm. The SHA-1 algorithm [NIST, 1993], in Table 1, is required in the XML Signature specification. The only parameter is the element to be signed. The provider of the Web service receives the message, the digest, and is told which algorithm has been used to create the digest. Using this information, the service provider is able to recreate the digest and compare it to the digest which it received from the requester.

When a message is passed from a requestor Web service to a provider Web service, the message body should be digitally signed using the XML Signature specification. There are several Token options for signing a message. These options fall under one of two categories; they can either be endorsed or unendorsed. An endorsed token is one which the claims of the Token can be validated by a trusted authority. An example of this kind of Token is a X.509 certificate. An unendorsed Token is one which the claims may not be validated by a trusted authority. However, there is such a thing as a proof-of–possession unendorsed Token. An example of this is a username-password Token.

When signing a message, the signature parameters consist of a security token and the message digest. It is worth noting that the second parameter can be an XPath node-set. The output is the message signature which will appear in the message header. Figure 2 [WS-Security, 2002] shows the key or token used to sign the message (1), the message digest (2), the signature value (3), and the unencrypted message body (4).

The provider service must have a way to verify the contents of the message. In order to do so the provider must have the message, the digest, determine the algorithm used to create the digest, and access the key or token. Security elements in the header of the message contain information on the algorithm and Token. The provider can use this information to compare the digest to the message. Any changes, even the addition of one white space to the original message can be detected. This clearly solves the problem of Message Level Protection.

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope
        ....
   <S:Header>
      <m:path xmlns:m="http://schemas.xmlsoap.org/rp">
         .....
      </m:path>
1     <wsse:Security>
<wsse:BinarySecurityToken ValueType="wsse:X509v3"         EncodingType="wsse:Base64Binary"
Id="X509Token">
                 MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
         </wsse:BinarySecurityToken>
         <ds:Signature>
            <ds:SignedInfo>
               <ds:CanonicalizationMethod Algorithm="http://....#"/>
               <ds:SignatureMethod Algorithm="http://...."/>
               <ds:Reference>
                  <ds:Transforms>
                     <ds:Transform Algorithm="http://...#RoutingTransform"/>
                     <ds:Transform Algorithm="http://.....#"/>
                  </ds:Transforms>
                  <ds:DigestMethod Algorithm="http://...#sha1"/>
            2     <ds:DigestValue>EULddytSo1...</ds:DigestValue>
               </ds:Reference> </ds:SignedInfo>
        3   <ds:SignatureValue>
              BL8jdfToEbl1/vXcMZNNjPOV...
            </ds:SignatureValue>
            <ds:KeyInfo>
                <wsse:SecurityTokenReference>
                    <wsse:Reference URI="#X509Token"/>
                </wsse:SecurityTokenReference></ds:KeyInfo></ds:Signature>
</wsse:Security></S:Header>
   <S:Body>
      <tru:StockSymbol xmlns:tru="http://..../payloads">
4       QQQ
      </tru:StockSymbol>
   </S:Body>
</S:Envelope>
```

Figure 2 – Soap Message with XML-Signature

2.3.3 Message Validity

Message Validity is ensuring that the contents of a message are appropriate to the service and that they are well formed. Checking the contents of a message can be subdivided into two categories, verifying data types and checking for malicious code. Verifying that the data types passed to an operation are those which the services are expecting is straight forward. Checking for malicious code within the message is not so straight forward.

Malicious code within a message can appear as part of the XML message or as parameters to be passed to operations. XML viruses and XML worms are commonly passed within the contents of any XML document or message [Lilly, 2002]. Because these are common in the Web environment there is software available to safely scan XML to determine if it contains either a virus or a worm.

Even after verifying that the parameters within a message are appropriate for the operation(s), their may be malicious code present. For example, it may be verified that a string is being passed to an operation which then queries a SQL database. SQL injection attacks are of the string data type. Therefore verifying that a string is being passed is not enough. Best practices for programming disallow and check for the presence of a ‘;’ in any parameter which will be passed to a SQL database. The ‘;’ in SQL allows for SQL commands to follow.

Ensuring that a message is well-formed is another step in Message Validity. Since the messages are in XML, it is possible that a message contains a circular-reference. A circular-reference may appear maliciously or through poor programming. Circular-references cause a system to encounter a run-out-of-memory error and shutdown [Lilly, 2002]. When done maliciously this is know as a denial-of-service attack. Proper parsing of a message will catch nested loops.

2.3.4 Authentication

Authentication can easily be described as verify to ones own level of certainty that an entity is who they claim to be. In its simplest form, authentication could be a username and password combination. However, this is only possible if there is already a relationship between the requester and provider.

Because of the distributed nature of Web services, a requester may be previously unknown to the provider. When an unknown requester authenticates it sends information about themselves to the provider. This information is known as credentials. It is up to the provider to verify this information. Now there are different degrees of verifying credentials and this can be directly affected by the type of credential that is sent. This is where the provider's own degree of scrutiny comes into play. In general, the more sensitive the information is which is being made available through a Web service, the higher the level of certainty must be. This certainty can be achieved through verification of the credentials. In the case of a previously unknown requester, the highest level of certainty can usually be achieved through a trusted authority. Trusted authorities issue certificates which can be used for authentication. A provider can evaluate the certificate and contact the trusted authority for verification.

However, their may be an intermediate service contacting the provider on behalf of the requester and once established the requester and provider will communicate. Assuming that the intermediary has authenticated the requestor and there is a trust relationship between the intermediary and the provider, the provider may take the 'word' of the intermediary and believe with a level of certainty that the requester is whom they claim to be. This can be done through the use of SAML or a certificate issued by the intermediary. Here the intermediary is providing the verification.

2.3.5 Authorization

In organizations, highly sensitive data and information must be protected with access control systems. These control systems allow defining and controlling which users are authorized to access specific applications and data but prohibit the access of unauthorized users.

Nowadays, organizations are built on heterogeneous IT infrastructure. As a result, a variety of systems with proprietary access control mechanisms, such as Unix, Windows, MAC, and mainframes exist and are incompatible. In proprietary access control systems, information about resources and attributes is stored in repositories called Access Control Lists (ACL). This is a problem since different proprietary systems have different ACL implementations, making it difficult to exchange and share information between them.

Authorization is the granting of rights, which includes the granting of access based on access rights. This typically takes place after authentication. Authorization is often confused with authentication, however it is a separate issue altogether. An access control implementation compares access control information such as the rights of the requestor with the policies or permissions needed to access the resource. If the rights of the requestor dominate the control policy, then access can be granted; otherwise access is denied. The two most common access control implementations are Access Control Lists (ACL), and RBAC. ISO 10181-3 specifies access control information used in making access control decisions.

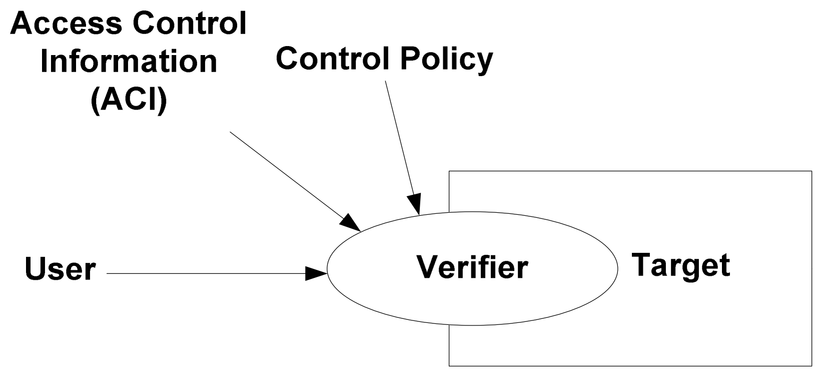


Figure 3 - Access Control

2.3.5.1 ACL's

ACL's are often used in the Unix environment for file and directory security. Although ACL's offer much more granularity than pervious *nix access control mechanisms, they can be cumbersome to implement and manage. There are difficult to manage because of the lack of relationships between the access control entities, i.e., resources, permissions, groups, and users. There is an obvious relationship between users and groups, users belong to groups and groups contain users. However, each shared resource must have an ACL file specified for it and the associated permissions are held within the file.

Users within groups can easily be managed, but for resources that change frequently like those in Web services it is difficult to modify the ACL's for all these resources. Therefore management of a ubiquitous and dynamic resource environment is cumbersome at best. Furthermore, performance is affected each time ACL is accessed and inspected. A simple example ACL is given below in Figure 4.

```
# file: documents
# owner: somebody
# group: other
user::rwx
user:jackson:rwx  #effective:rwx
user:smith:rwx                  #effective:rwx
group:publ:rw-                  #effective:rw-
mask:rwx
other:---
```

Figure 4 - ACL

2.3.5.2 RBAC

In 2004 the National Institute of Standards and Technology (NIST) published a standard [NIST, 2004] for defining the features of Role Based Access Control. The standard was largely based on the various features found in commercial implementations of RBAC. There are two parts to an RBAC system. The first is the Reference Model which consists of objects, operations, permissions, roles, and users. The second is the System and Administrative Functions which include system functionality, and administrative operations and reviews [NIST, 2004]. Our approach utilized the concepts of RBAC, so we will discuss it here. However, much has been written about RBAC over the past decade; in an effort not to be repetitive this is a summary-review.

RBAC contains Permissions sets. Generally speaking, Permissions express a privilege to access a resource. Permissions are a set of one or more objects and one or more operations. Objects refer to resources; for example a printer or a file. Operations are the invocation or execution of some function on a resource. An example of a Permission may be “create file in directory etc’. In this case, ‘etc’ is the Object and ‘create file’ is the operation.

Once Permissions are created, they may be assigned to Roles. A Role is a job function which is performed within an organization. A job function can be as concrete as a job title, ‘Physician’, or more general even abstract, ‘internet user’. Once Roles are defined they may be assigned to Users who are actual people. Users may also include entire organizations, computers or networks [NIST, 2004].

The HL7 committee has developed a simple and effective way for creating Permissions (Object, Operation set) and Roles. They call it a scenario driven approach. The concept is to first create scenarios for the ‘organization’. These scenarios include resources, actions taken on the resources, and who is performing these actions, in terms of job function. Our Detailed Scenario is an example of this approach.

Figure 5 shows the relationships between the elements of the RBAC Reference Model. Permissions are an Object, Operation set. There is an assignment between Permissions and Roles, and Roles and Users. These relationships/assignments are many-to-many. There is another element present in the figure which has a one-to-many assignment with Users. That element is the Session Role [NIST, 2004].

A Session is the activation of one or more Roles by a User. Simplistically and not entirely, a Session Role determines if a Users Role should be activated. This is determined by the constraints on the Roles assigned to the User and which Roles the User currently has active. The RBAC Reference Model provides fine granularity for authorization of resources. The RBAC Systems and Administrative Functions provide for distributed decision and enforcement points.

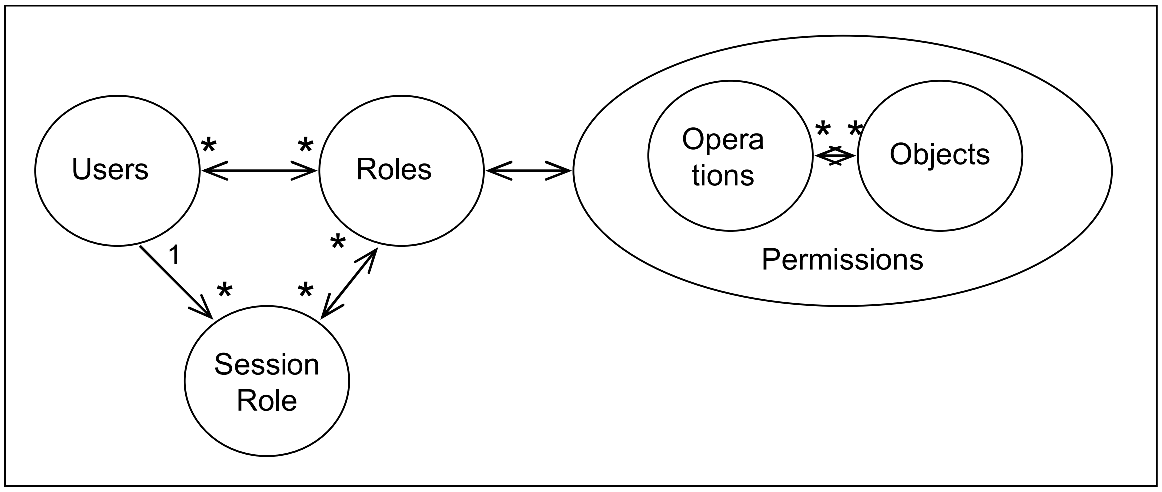


Figure 5 - RBAC

In our approach we conceptually map elements of RBAC to elements of Web services in order to an authorization function regarding the Web service and a prediction function regarding the client. In or approach, RBAC Operations are mapped to the action that an operation of a Web service performs. Keeping with our example, the Web service has an operation to review the medical history of a patient; this operation is mapped to the RBAC element 'read'. The RBAC element Object is mapped to the resource which the service accesses and the parameters of the operation; i.e. the medical history of the patient and the patient respectively.

We define an authorization function, and as mentioned in section 1.2, there is a variable of uncertainty between what a Web service provider is describing and what a client may assume the provider is describing. This variable is caused because our ontology is an upper level ontology; therefore each organization using the ontology must map Users, Roles, and possibly Groups to the ontology. This mapping is independent of any other organization; therefore there is uncertainty that a client and service provider have the same mapping. Our goal is to minimize the variable to achieve accurate results.

We define the authorization function as $f{:}{<}UA, S, O, P_{i,} \ldots, P_k) \rightarrow \{0,1\}$ where S is the service, O is an operation, P is a parameter and UA is defined as $UA \subseteq U \times R$. UA is the many-to-many user-to-role mapping relationship within the RBAC ontology. However, to predict authorization we use UA' which is a many-to-many mapping between UA and $(U' \times R')$, the many-to-many user-to-role mapping within an organization. Therefore, UA' is defined as $UA' \subseteq UA \times (U' \times R')$, the many-to-many mapping between the ontology and the organization structure. So our prediction function is defined as $f{:}{<}UA', S, O, P_{i,} \ldots, P_k) \rightarrow \{0,1\}$

Structural Role Framework

Structural roles serve as access control decision information within the PMI (Privilege Management Infrastructure) by allowing authenticated users to establish a session or connect to a protected target.

To accomplish this function, the user asserts, in addition to authentication information, structural roles as a prerequisite authorization to “connect” to the task or workflow containing the requested session or target. An infrastructure access enforcement function grants or denies access to the session or target based on the structural role. Structural roles would be typically managed in identity certificates (per ASTM E2212) or directories. Structural roles are centrally managed, allowing any user to be granted access, suspended, or denied access (by means of the service-oriented Verifier) to any or all resources through this single point of control.

ASTM E1986 identifies healthcare persons for whom role based access control is warranted. These ASTM E1986 person types define basic healthcare role names as used within this standard.

Functional Role Framework

Functional role activation (session roles) cannot occur until the session is established, so structural role authorization/access is prerequisite to establishing a session or connection to the target. In the extended Control Model, what is desired is a decision on the user's authorizations to perform operations on the Target's protected objects. The result of the decision information is used as an input to the Verifier PEP (Policy Enforcement Point) for the purpose of access control.

Functional roles describe the permissions that a user has available once the session is established and his/her roles are activated. Functional roles are contained in applications, directories, attribute certificates, and XACML extensions. Functional roles specifically define, in terms of permissions, what authorizations are needed by an entity to access protected information technology or application resources. As a consequence, functional roles are much more healthcare specific than basic roles. Standard functional roles are applied across the enterprise and with business partners. Standard functional roles are aligned by mapping to underlying applications' enforcement mechanisms.

Functional roles should be expressed in a standards-compliant language for interoperability both inter and intra-enterprise. A standard language allows for leveraging policy and roles among applications, as well as consistent policy description and enforcement. The guideline standard language for this standard is OASIS XACML. In an ontology, it is necessary to use first order description logics to express these complex relationships. We will discuss this further in section 6.3 and section 8.

2.3.5.4 XACML

XACML is an OASIS standard for an XML representation of RBAC [XACML, 2005]. The Organization for the Advancement of Structured Information Standards (OASIS) standards group developed the eXtensible Access Control Markup Language (XACML) as a language to express and evaluate access decisions.[i] The XACML technical specification includes a profile for RBAC using XACML that complies with the ANSI RBAC standard. Core RBAC, as defined above, is supported as shown in Table 2 below.

Core Element	XACML Profile
Users	XACML Subjects
Roles	XACML Subject Attributes
Objects	XACML Resources
Operations	XACML Actions
Permissions	XACML Role <PolicySet and Permission <PolicySet>

Table 2 RBAC Core Functionality Mapping

The XACML RBAC profile also supports hierarchical RBAC, allowing inheritance between roles. Dynamic Separation of Duty is supported by the profile, and structural Separation of Duty can be supported via the user-role assignment mechanism. Additional XACML policies are provided to support system and review functions described in the ANSI RBAC standard. Specifically, the Role PolicySet (RPS) associates holders of a given role attribute with a Permission PolicySet. The Permission PolicySet (PPS) describes the permissions associated with a specific role. The RPS and PPS replace the role assignment and role specification ACs in the X.509 based role model.

The XACML role based PMI features a rich policy language integrated throughout the design. The concept of structural versus functional roles is supported using a two tiered system comprised of a role attributes. That is, users can have roles assigned to them in the request context. An entity separate from the policy decision point can use an XACML Role Assignment Policy or PolicySet to enable attributes within the user session.

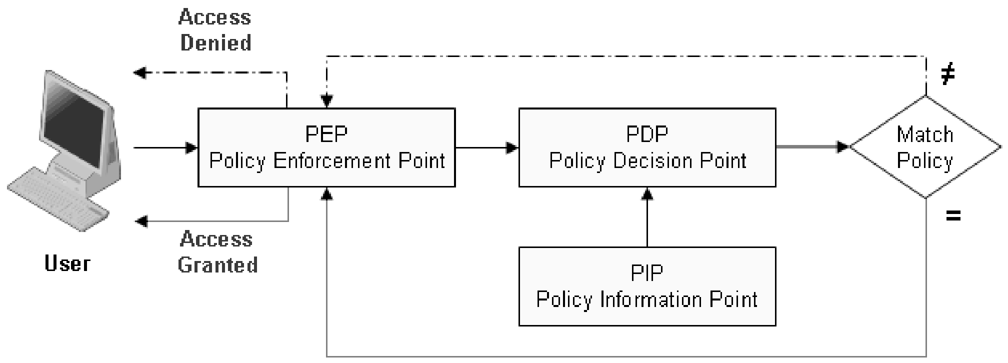

Access Denied
User
Access Granted
PEP Policy Enforcement Point
PDP Policy Decision Point
PIP Policy Information Point
Match Policy
≠
=

Figure 6 - Typical XACML usage scenario

Figure 6 illustrates a typical XACML usage scenario. A subject (e.g., human user, application) wants to take some action on a specific network resource, such a file system or Web service. The subject submits its request. The request for authorization goes to the entity protecting the resource, the PEP (Policy Enforcement Point). The PEP uses XACML request language to create a request based on attributes of the subject, action, resources and sends it to the Policy Decision Point (PDP), which evaluates the request. The PDP invoke the Policy Information Point (PIP) service to retrieve applicable policies written in XACML that are applicable to the request. The PDP compares the request against policies and determines whether access should be granted according to the XACML rules for evaluating policies. Policies contain information about the subject, the action, and other environmental properties. The result of the comparison can be either access granted or denied. The answer goes back to the PEP. If there is no match, the PEP denies user access; otherwise, it permits access by the user.

While there are many proprietary languages for controlling the access to resources, XACML has advantages. The use of a standard access control policy language can replace several proprietary languages making easier the interoperation of applications. Programmers and administrators can work more efficiently since they do not have to develop new policy languages and write code to support them and only need to understand one language. The use of a common language allows one policy to be used by many different applications, thus making policy management easier. Policies can also be distributed by referring to other policies stored in geographically disperse locations. For instance, a local-specific policy may refer to an organization-wide policy.

2.3.5.5 WS-Authorization

WS-Authorization is a proposed future specification regarding the description and management of authorization data and policies [IBM, Microsoft 2002]. In particular, WS-Authorization will specify a standard on how to describe authorization claims within a security token and how the end-point should interpret these claims. It is widely thought that this specification will be a follow-on specification to WS-Privacy and WS-Security, as seen in Figure 7. WS-Privacy and WS-Security are implemented in WS-Policy, and it is believed that WS-Authorization will be implemented in WS-Policy as well. It is also believed that this specification will be similar in structure to the XACML standard [Rosenberg, Remy, 2004].

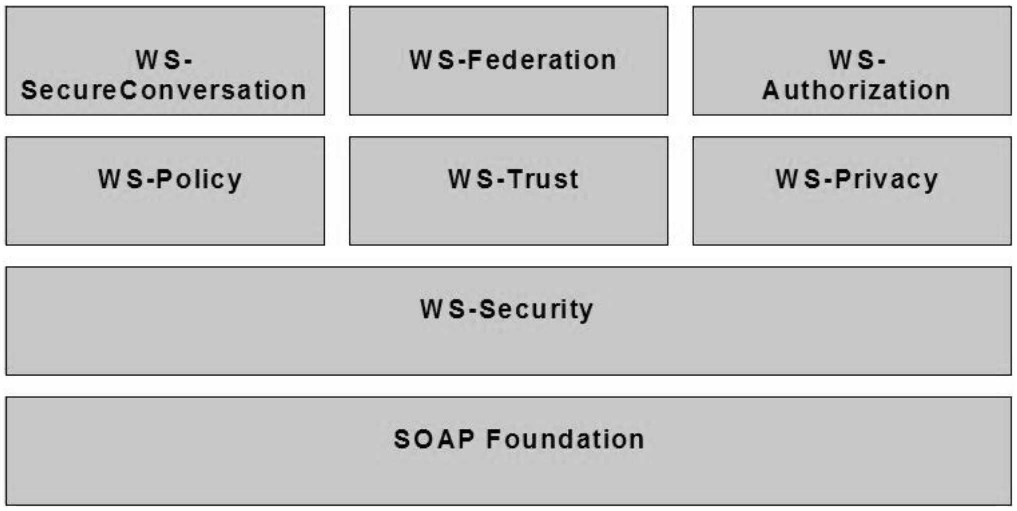


Figure 7 - Web services stack

Should the WS-Authorization specification be similar to the XACML standard, this would validate our approach of using RBAC and XACML concepts to annotate WS-Policy files. Further, IBM and Microsoft concur that the end-point policy files are the appropriate location for describing “execution capabilities” [IBM, Microsoft 2002] of an authenticated client. However, we have seen little evidence of incorporating semantics into WS-Authorization to aid in the discovery of services which a client will be authorized to invoke or execute. By using ontological concepts to describe execution capabilities of an authenticated client, a potential client can automate the prediction of their authorization.

2.4 Web Services Process

BPEL is an XML language designed to work with the Web service technologies WSDL, SOAP, and UDDI. Its purpose is to provide a way to describe a business process, or workflow, in the context of the Web services environment. A business process is made up of one or more tasks. Each task can then be represented as a Web service.

There are two types of BPEL processes that can be created. The first is the Abstract process which is like a template. The Abstract process contains information about the input and output parameters, and what each task should accomplish. The Abstract process does not specify any bindings and links which concern how the process will be executed.

In an Executable process, all details concerning the process are present. This includes flow control, links, and bindings, along with the information found in an Abstract process. Executable BPEL processes are said to follow the Orchestration paradigm. BPEL allows for describing the process of the flow control in two ways, either sequentially or in parallel.

The usefulness of BPEL is in the execution of an entire process. It is the end result of the process which is the objective of creating a BPEL process. That said, authorization must be granted for the entirety of the process. The primary concerns in predicting authorization for a process are cross domain interoperability with respect to authorization, and the flow of the process in which authorization to a service is dependent on authorization being granted from another service.

To circumvent these two possible pitfalls we take a granular approach. Prior to our constraint analysis, discussed in section 6.4, we construct a client policy for each task associated with a process. These policies are constructed from the BPEL process policy, any associated BPEL task policies, and any policy files associated with the Semantic Template, discussed in section 6.2. Our constraint analysis then follows the flow of control established in the BPEL process. In order for a candidate set of services to be considered for execution it must pass the constraint analysis as outlined throughout section 6.

2.5 Ontology HL7 – RBAC

Our ontologies are represented in OWL-DL [OWL, 2004], *Web* Ontology Language - Description Logics. We started with two separate Upper Level domain ontologies, a RBAC ontology and a HL7 ontology, as seen in Figure 8. We then created the HL7 RBAC ontology by expanding the RBAC Upper Level ontology through the use of the HL7 RBAC Permissions Catalog [HL7 Security Technical Committee, 2005]. This catalog contains operations and objects which have been paired together to form permissions. For other domains in which an RBAC standard is not available, concepts from a domain may be imported into the RBAC ontology in order to create a domain specific Mid Level ontology.

In addition, we have used a comprehensive list of medical departments [Hull and East, 2006], and a list of the 31 broad industry categories. According to the NAICS (North American Industry Classification System) it would be advantageous, for security reasons due to cross-classification of small industries, to ultimately expand the industry portion of our Ontology to include the 265 detailed categories [Census, 200].

We developed two more specialized Lower Level ontologies, one for the client and one for the service provider, by extending our mid level ontology in an effort to more accurately model the real world. The client ontology was developed to reflect an organizational implementation. This was done by adding users, assigning them to roles, using variations of the role names, and assigning appropriate permission to these roles. The service ontology was extended in much the same way with the exception of adding users, which is practical for security reasons. The fundamental difference between the ontologies is variation of role to permissions assignment, as well as role names. We included some different role names, between the ontologies, for the same job function. For example, ‘Radiology Technician’ and ‘Radiology Tech’, ‘General Physician’ and ‘Family Practice Physician’, and ‘Pediatric Nurse’ and ‘Pediatric RN’. The above titles are all standard titles for positions in the Health Care domain. Since many organizations had implemented systems using few variants of position titles, the most common variants were selected into the standards mentioned above.

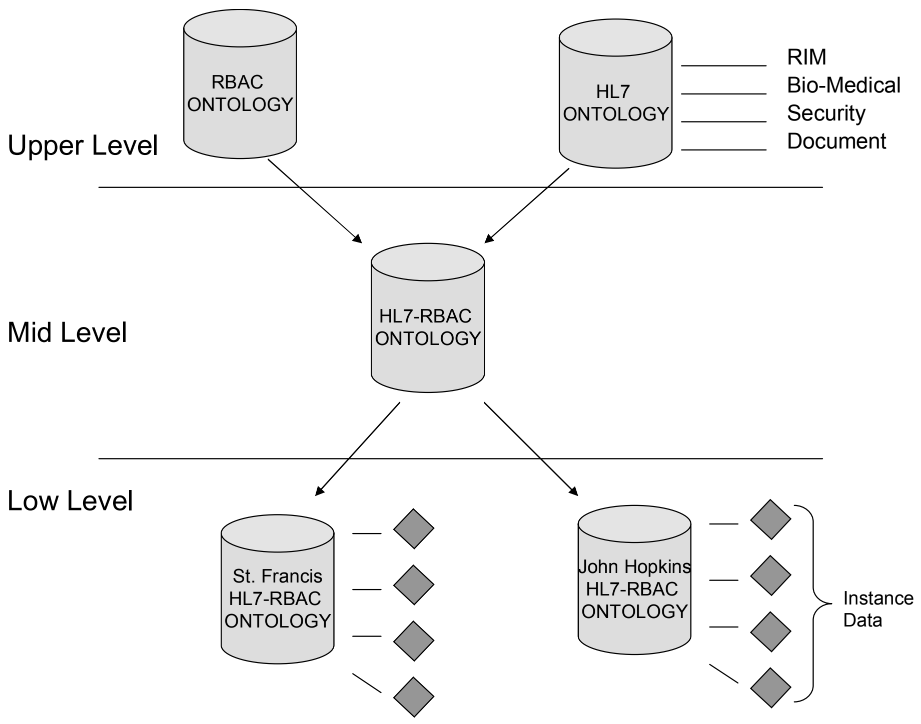
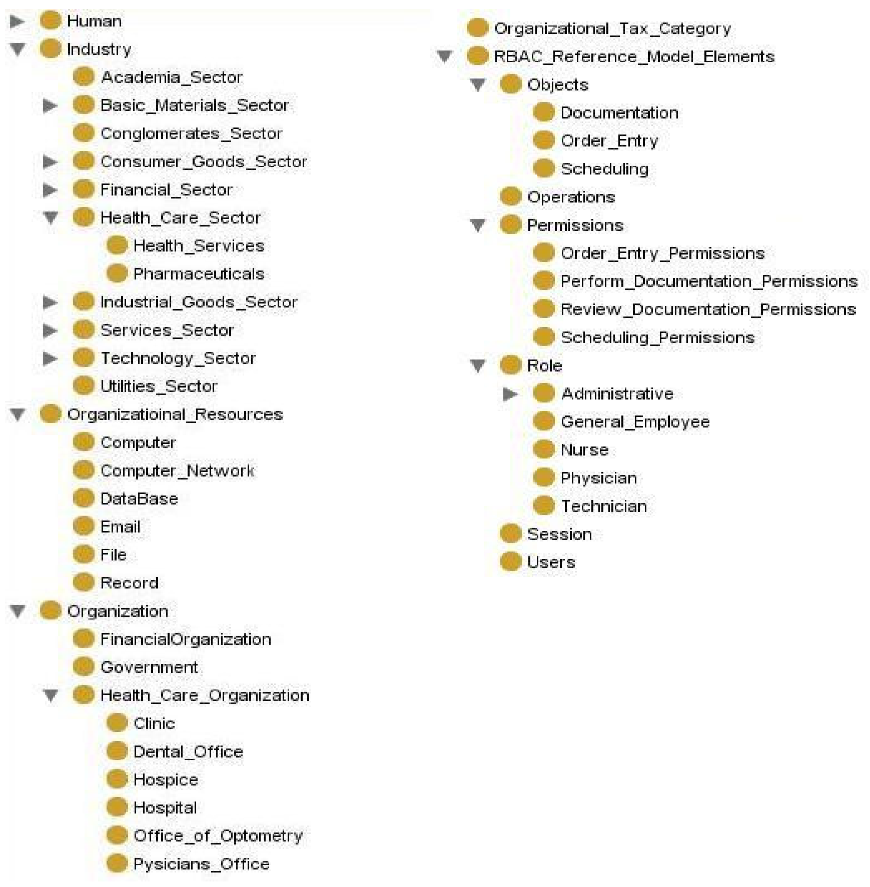


Figure 8 - HL7 & RBAC Ontology Hierarchy

Figure 9 below shows a portion of our HL7 RBAC Mid Level Ontology. The ‘RBAC Reference Model Elements’ is the parent to the actual elements, namely Objects, Operations, Permissions, Role, Session, and Users. There are relationships between these elements, more accurately ontological concepts. In section 6.4.1, we discuss how we use these relationships. Let us now describe some of the relationships.



- Human
- Industry
  - Academia_Sector
  - Basic_Materials_Sector
  - Conglomerates_Sector
  - Consumer_Goods_Sector
  - Financial_Sector
  - Health_Care_Sector
    - Health_Services
    - Pharmaceuticals
  - Industrial_Goods_Sector
  - Services_Sector
  - Technology_Sector
  - Utilities_Sector
- Organizatioinal_Resources
  - Computer
  - Computer_Network
  - DataBase
  - Email
  - File
  - Record
- Organization
  - FinancialOrganization
  - Government
  - Health_Care_Organization
    - Clinic
    - Dental_Office
    - Hospice
    - Hospital
    - Office_of_Optometry
    - Pysicians_Office
- Organizational_Tax_Category
- RBAC_Reference_Model_Elements
  - Objects
    - Documentation
    - Order_Entry
    - Scheduling
  - Operations
  - Permissions
    - Order_Entry_Permissions
    - Perform_Documentation_Permissions
    - Review_Documentation_Permissions
    - Scheduling_Permissions
  - Role
    - Administrative
    - General_Employee
    - Nurse
    - Physician
    - Technician
  - Session
  - Users

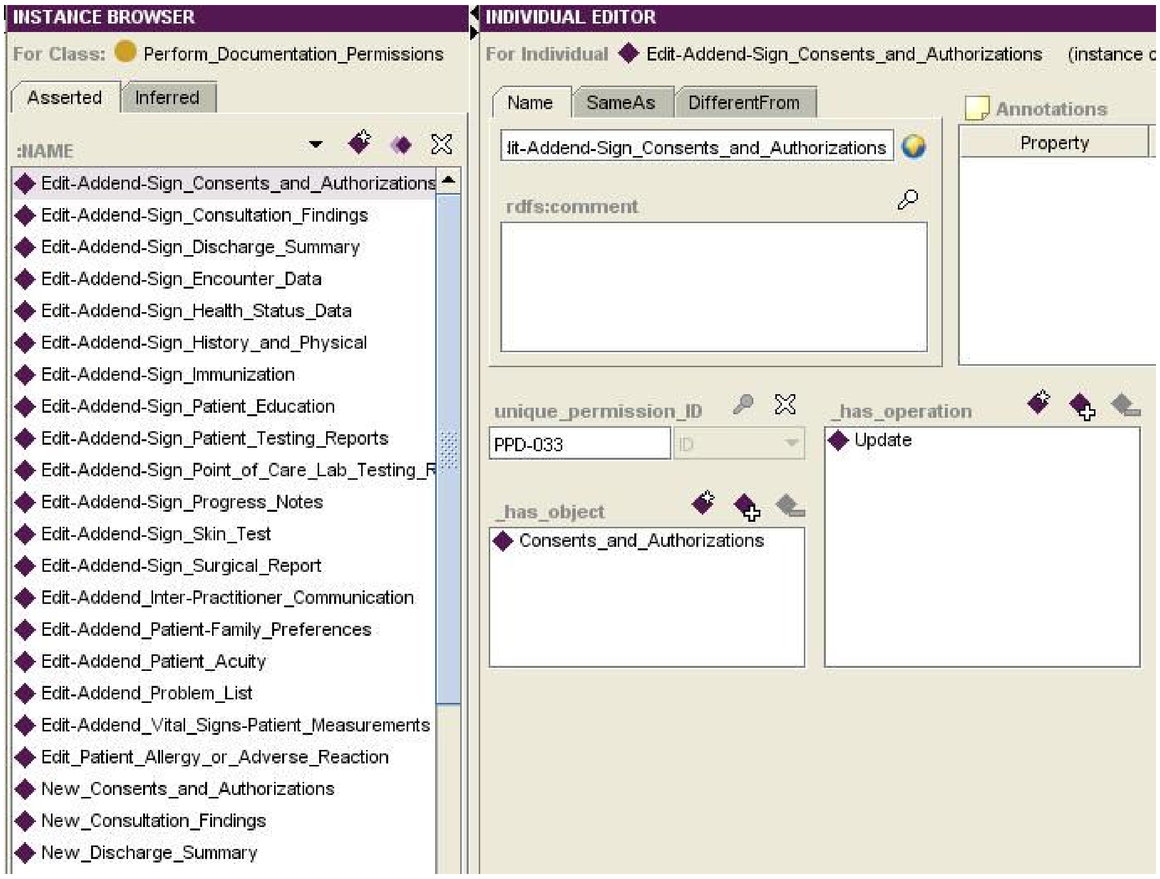
Figure 9 - Classes in the RBAC Ontology

Figure 8 above shows a portion of our HL7 RBAC Mid Level Ontology. The ‘RBAC Reference Model Elements’ is the parent to the actual elements, namely Objects, Operations, Permissions, Role, Session, and Users. There are relationships between these elements, more accurately ontological concepts. In section 6.4.1, we discuss how we use these relationships. Let us now describe some of the relationships.

As we stated earlier and as shown in Figure 10 below, permissions have a ‘has object’ relationship with Object and a ‘has operation’ with Operation. A Role has a ‘assigned to’ relationship with User, a ‘department’ relationship with Health Service (which is not visible in the figure), ‘has permission’ relationship with Permission, and so on. A User has a ‘assigned to’ relationship with Role, a 'employed by' relationship with Organization (which is not visible in the figure), a 'isA' relationship with Human, and so on. We will discuss more on the relationships in our example in section 6.4.1.

As can be seen below, there are many instances of permissions. These are all from the HL7 Permissions specification. This is not an exhaustive list of the possible permissions; rather the committees' goal was to provide a general starting point which provides examples so that health care organizations could correctly create permissions tailored to their organization.

The HL7 RBAC ontology is available for download from the LSDIS Web site at http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/ontologies/HL7_RBAC.owl.



For Class: Perform_Documentation_Permissions

- Edit-Addend-Sign_Consents_and_Authorizations
- Edit-Addend-Sign_Consultation_Findings
- Edit-Addend-Sign_Discharge_Summary
- Edit-Addend-Sign_Encounter_Data
- Edit-Addend-Sign_Health_Status_Data
- Edit-Addend-Sign_History_and_Physical
- Edit-Addend-Sign_Immunization
- Edit-Addend-Sign_Patient_Education
- Edit-Addend-Sign_Patient_Testing_Reports
- Edit-Addend-Sign_Point_of_Care_Lab_Testing_R
- Edit-Addend-Sign_Progress_Notes
- Edit-Addend-Sign_Skin_Test
- Edit-Addend-Sign_Surgical_Report
- Edit-Addend_Inter-Practitioner_Communication
- Edit-Addend_Patient-Family_Preferences
- Edit-Addend_Patient_Acuity
- Edit-Addend_Problem_List
- Edit-Addend_Vital_Signs-Patient_Measurements
- Edit_Patient_Allergy_or_Adverse_Reaction
- New_Consents_and_Authorizations
- New_Consultation_Findings
- New_Discharge_Summary

For Individual Edit-Addend-Sign_Consents_and_Authorizations (instance c

lit-Addend-Sign_Consents_and_Authorizations

unique_permission_ID: PPD-033

_has_operation: Update

_has_object: Consents_and_Authorizations

Figure 10 - Instances example

2.6 Prototype Implementation

Let us start with stating our goal, then moving on to a description of our approach and then looking at an overview of the components. Our goal is to add authorization analysis to the Semantic Discovery of Web services. Our approach is to add semantics to WS-Policy, regarding authorization, which will be used by the client to determine if they have authorization to access the provided resources. Implementation of our approach is done on the Server side and client side.

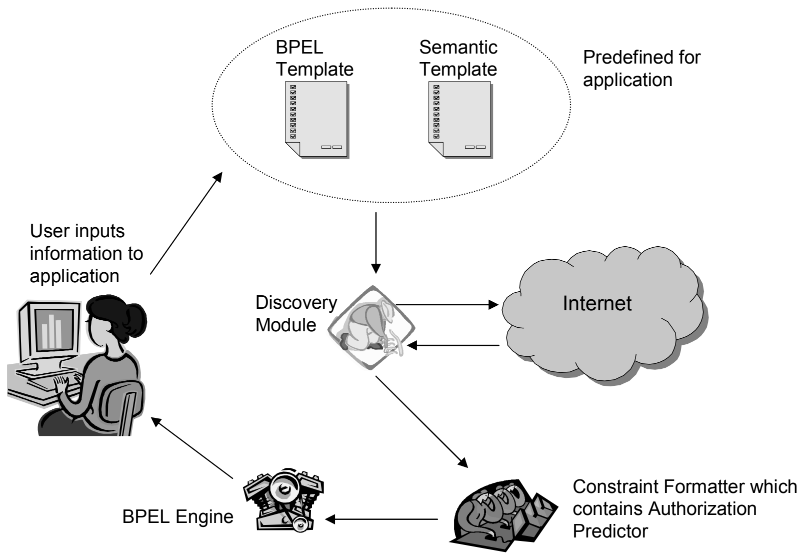

BPEL Template
Semantic Template
Predefined for application
User inputs information to application
Discovery Module
Internet
BPEL Engine
Constraint Formatter which contains Authorization Predictor

Figure 11 - Flow of Information

Here we will introduce the flow of information from a big picture perspective. We will expand on our approach starting with the system architecture in sections 6.1. In section 6.2, we will discuss the WS-Policy semantic annotations. Authorization Prediction, which takes place on the client side, is discussed in section 6.3.

From our running example, an Emergency Room Physician at St. Francis hospital wants to review the Medical History of a patient. The Physician may enter information about the patient and themselves into an application, Figure 11. The information is passed into a predefined BPEL template, BPEL process WSDL and its attached policy file. A Semantic Template is created from the process WSDL file and passed to discovery. Once Web services are discovered, the information for those services is passed our Constraint Formatter. Once analysis and Authorization prediction is complete the selected web services are passed to the BPEL engine for execution, if it is a single service then it is invoked through a Web service client.

2.6.1 Architecture

The architecture is divided into three sections; Service Side, Global Resources, and Client Side. The Service Side contains the Web services. Authentication and authorization are implemented here and in our context are viewed as a 'black box'; this is because our framework does not require any specific implementation to be used. We have implemented a primitive "Username Token" authentication scheme for basic testing purposes. Authorization is then preformed on the "Username".

Global Resources include the UDDI for semantic discovery, the ontologies used for annotations, and the files associated with the published Web services, i.e., WSDL-S [Akkiraju et al., 2005] and WS-Policy. SAWSDL [SAWSDL, 2006], the next generation of WSDL-S, is currently a W3C specification submission which supports semantic representation in WSDL through annotations. We use a UDDI registry that is capable of semantically matching the requirements in a Semantic Template with those in the providers' WSDL-S files [Verma et al., 2004]. A Semantic Template is an abstract representation of the requirements for a service, describing the functional requirements of the service using ontological concepts [Fensel et al.] [Dogac et al., 2002].

The Client Side contains necessary information for discovery, Authorization Prediction, and execution of the BPEL process. As mentioned above, discovery will use the Semantic Template(s) created by the client. Authorization Prediction uses information from the BPEL process policy file, ontologies, and the annotated WS-Policy file.

The non-functional requirements of a Web service requirements are specified in a policy file associated with each Semantic Template. As for the client authorization information, it is placed in the BPEL process policy file which is discussed in section 6.3.2 and can be seen in Figure 13.

We use BPEL to create an abstract process. Specifically, we are using ActiveBPEL [ActiveBPEL 2.0, 2006] because of the easy-to-use graphical user interface (GUI), ActiveBPEL Designer [Active Endpoints, 2006]. In HL7 there are many scenarios which have been created by the committees. These scenarios can be transformed into a BPEL process. The flow of the process is created using the GUI. The requirements of each task in the process are then described in a Semantic Template for each task; see Figure 12(1).

The client then sends the Semantic Template(s) to the discovery engine (Figure 12(2)). Lumina, a tool created at the Large Scale Distributed Information Systems Lab, at the University of Georgia, performs semantic matching of the functional requirements during the discovery phase 12(3).

Once the discovery engine has returned a set of candidate services for a given task (Figure 12(5)), Authorization Prediction is preformed through the use of Ontology inferencing (Figure 12(6)), which we discuss in section 6.4.1. Constraint analysis is preformed on non-functional requirements (Figure 12(8)). The most optimal and appropriate services are then bound to the abstract process (Figure 12(9)). Currently, binding the abstract process to services is done manually at process deployment time, though we are researching ways to automate binding at run time. Once the executable process is created it may be invoked. If it is a single service then we can invoke this through a Web service client.

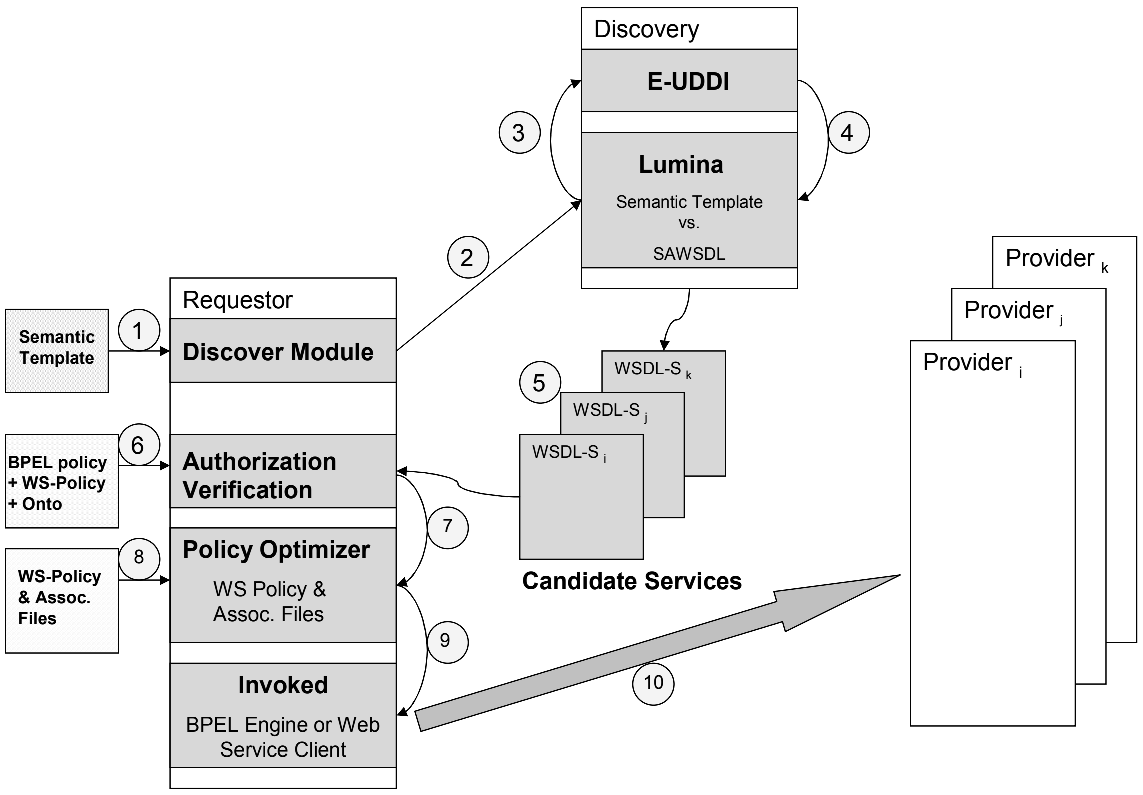


Figure 12-Information Flow

2.6.2 Server Side

WS-Policy is a Web services specification regarding how to describe the non-functional requirements of a Web service. A policy file is provided to the client, or requester, by the service provider, which is the server side. As we discussed in section 3.5.5, WS-Policy does not include information regarding authorization of clients at this time.

Our approach is to use ontological concepts to annotate a WS-Policy file. The annotations express what type of requestor has authorization to use the associated Web service; all else is an implicit deny. As mentioned earlier, these annotations can be linked to the WSDL file following the WS-Policy attachment specification. Next we will discuss the annotations which are extensibility elements.

It is important to note that we are not using the annotations to enforce access control; rather it is an aid to Semantic discovery.

2.6.2.1 Extension Elements

There currently is no a WS-Authorization specification. After reviewing current specifications that are built onto WS-Policy (WS-Agreement, WS-Transaction, WS-Security) we were convinced that a future WS-Authorization specification would follow suite and therefore lack the semantics needed for an automated process. There for in our development of a WS-Authorization specification we provided the option to incorporate semantics. By automating the predication of authorization, a client or consumer can save time and more efficiently allocate its resources. By designing semantics into the WS-Authorization specification, we have made the information machine interpretable, thereby reducing the need for human interaction. In addition, semantics provide relationship information about a concept which can be used to predict authorization. Lexical matching does not provide any of the benefits we mentioned.

The authorization annotations are extensibility elements similar to the extensibility elements provided in WSDL-S [Akkiraju et al., 2005], for example precondition and effect. The annotated

WS-Policy file, called a SemPolicy in [Verma, 2006], will provide extensibility elements for semantic representation of authorization. The extensibility elements are derived from the RBAC standard [NIST, 2004] and XACML representation of RBAC.

RBAC was chosen because it is widely accepted, easily understood, and succinctly expresses authorization permissions. However, the enforcement point for authorization does not need to be implemented using RBAC. It can be implemented in anyway suitable to the provider of a Web service. For example, in a Novell environment there may be a group named 'Doctors' which users who are doctors will belong. As a note, in an enterprise environment it is not uncommon for users to belong to more than one group. It should be clear to see that there is a one-to-one mapping from the group 'Doctors' to the ontological RBAC concept 'Physicians'. There may be a many-to-one mapping. For example, suppose a small to mid size organization does not make a distinction between types of doctors. There may be a mapping from the group 'Doctors' to 'Surgeon', 'OBGYN', 'Oncologist', etc.

Here we will cover the extensibility elements, their descriptions, and give examples. Our first extension element is *permission*. *permission* is the operation an authenticated client is authorized to perform on a certain object, which is a resource.

The extension element *role* is a function within the context of an organization; some associated semantics regarding the authority and responsibility are conferred on the user assigned to the *role*. A role could be general, for example 'Employee', or more specific as in 'Radiologist'. As well, a *role* could be a group which confers authority and responsibility as in 'Hospital Executive'.

Since it is not feasible to name each user and the semantics of a subject can vary greatly we use an extension element *subjectCategory* to describe the type of a subject. Subjects can be users, which the National Institute of Standards and Technology define as a human, machine, network, or intelligent autonomous agent. In our context, this can also include an entire organization. Using *subjectCategory* as an extension element enables us to describe relationships. For example, the *subjectCategory* partner describes that the subject is in some kind of partnership agreement with the provider of the Web service.

Lastly, *modelReference* is used to handle the mapping of a schema element to an ontological concept. For example, this can be applicable when a Web service provider wants to demonstrate that authorization will be constrained to certain inputs for an operation. This might be done using an ontological concept like patient_identification_number.

Our annotation framework provides the granularity needed for Web services. This is because a WS-Policy file may be attached to a message, a service binding, an operation, or a parameter such as an input. The annotations are used to describe an explicit 'grant'; while we assume lack of the criteria or conditions is an implicit 'deny'.

The WSP-S is an annotated WSP. As seen in Figure 13, annotations can occur after the <All> tag in WSP. If there is one annotation for the entire WSP then it could be placed after the first <ExactlyOne> tag. The first annotation in Figure 13 describes authorization for a requestor whose *role* is "Emergency Room Physician". The second annotation is a *subjectCategory,* namely "Health Services". From the namespace it is seen that the concepts are from the HL7 RBAC ontology.

```
<wsp:Policy
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:base="http:/http://www.NationalEHR.com/policies"
xmlns:wsrm="http://schema.xmlsoap.org/ws/2004/03/rm"
xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau"
xmlns:Ontology1="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/ontologies/HL7_RBAC.owl">
  <wsp:ExactlyOne>
   <wsp:All>
   <wsau:role name="Emergency_Room_Physician" wsau:ModelReference="Ontology1# Emergency_Room_Physician "/>
   <wsau:subjectCategory name="Health_Services" wsau:ModelReference="Ontology1#Health_Services"/>
   <wsse:SecurityToken>
               <wsse:TokenType>
                              wsse:X509v3
               </wsse:TokenType>
   </wsse:SecurityToken>
   </wsp:All>
  <wsp:All>
   <wsau:permission name="read" wsau:ModelReference="Ontology1#read"/>
   <wsau:role name="Executive_Administration" wsau:ModelReference="Ontology1#Executive_Administration"/>
   <wsau:subjectCategory name="Health_Services" wsau:ModelReference="Ontology1#Health_Services"/>
  <wsse:SecurityToken>
               <wsse:TokenType>
                              wsse:X509v3
               </wsse:TokenType>
  </wsse:SecurityToken>
  </wsp:All>
</wsp:Policy>
```

Figure 13 – Annotated WS-Policy file

We allow for multiple annotations within the policy file. This enables a provider to express multiple conditions regarding authorization. For instance, an 'Emergency Room Physician' who is also affiliated with an organization that is categorized as 'Health Services' may have authorization to a providers' resource, while all other Physicians do not. This is accomplished by placing both annotations within the <ALL> tag. This can be seen in the example above.

There is also the situation in which a requester can have authorization to access a resource if it meets one condition or one set of conditions described by the provider. In this case the annotations are placed within the <ExactlyOne> tags. Figure 13 shows two sets of conditions with in the <ExactlyOne> tags. The authorization information from this figure can be read as authorization may be granted to someone that is an Emergency Room Physician that is affiliated with an organization in Health Services or an Executive Administrator who has read privileges and is affiliated with an organization in Health Service.

Any domain specific ontology can be used for the annotations. However, a quality of RBAC is that it has a structural hierarchy with relationships which lends itself to the creation of an ontology schema. The concepts of RBAC include organizational and professional roles. This fits well with the extension elements derived from the XACML representation of RBAC.

The annotations begin with the namespace "wsau" which is declared in XML declarations as follows: xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau". This is depicted in the previous figure. The xsd is available at "http://lsdis.cs.uga.edu/authorization/wsau.xsd".

2.6.3 Client Side Authorization Prediction

Once Semantic Discovery has returned a set of candidate services, the requestor performs constraint analysis to determine which of the candidate services it most likely has authorization to invoke, Authorization Prediction. Authorization Predication uses information given about the client, WSP-S, and ontologies to make the 'best choice'. Our approach is to have authorization information for the client contained in process WSDL's attached policy file, i.e. the BPEL process policy file. Our approach enables us to handle semantic discovery for both a BPEL process and a single Web service.

By placing the client authorization information in the BPEL process policy file we achieve two goals. First, the same authorization information is used for the entire process. This is important since each task must allow invocation for the process to complete. Second, this approach allows us to reuse a process for multiple users, only having to change the authorization information.

Authorization Predication begins by taking those services returned by the discovery engine in the Semantic Discovery phase to create sets of candidate services. Each set of candidate services satisfies the BPEL process functional requirements. For example, suppose a process has tasks (a) and (b); then each set has a candidate service (a) and candidate service (b).

Next we perform constraint satisfaction and constraint optimization analysis [Pennington, 2006]. During the constraint analysis process, if an authorization annotation is found then that information is passed to the Authorization Prediction manager. Information contained within the annotation, regarding the service provider, and information within the BPEL process policy, regarding the client, is used for ontology based inferencing to predict if the client has authorization to use the resource. If an authorization path is detected via queries then a relationship(s) exists between the concept(s) in the policy file and the concept(s) in the BPEL process policy file. The RBAC standard provides a defined structure such a path exists only if the concepts are related in such a way that authorization should be granted. Therefore, we would predict that authorization will be granted. If a path is not detected then we move to our second phase.

In the second phase, we try to determine if two concepts from different ontologies are equivalent. Even in a highly standardized domain such as Health Care, two concepts may have different names. For instance, an 'Emergency Room Physician" from one ontology may correspond to an "Emergency Physician' from another ontology, or "ER Doctor" or ER Physician". Therefore in our 'query phase' we would not determine the 'sameAs' relationship unless it previously existed in the ontology.

Our approach is to examine the relationships of the client and service concepts and those related concepts that are linked by these relationships. Because of the structure of the RBAC ontology, it is manually possible to quickly determine which relationships are most important. In most cases it will be necessary to place weights on the relationships in order to improve the accuracy of the results. This approach is based on [Dong et al., 2005] and [Aleman-Menza et al., 2006]. However, applying weights requires a human to review the relationships within the ontology. Our approach to the problem of weighting the relationships is as follows.

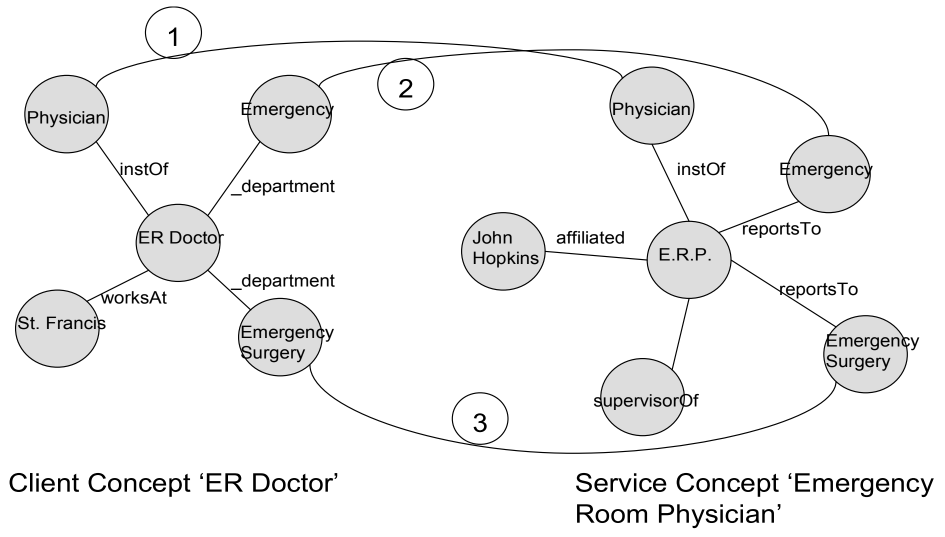


Figure 14 - Concept Comparison

At each iteration of our algorithm, we compare a concept related to the original service concept with one related to the original client concept. For example, in Figure 14 a the first iteration we compare the Physician concept related to ER Doctor from the client ontology to a concept related to Emergency Room Physician(ERP) in the service ontology. This continues and the concept Physician related to ERP is found, Figure 14 (1). As shown in the figure above, after our algorithm terminates, we have found three concepts related to ER Doctor that are similar, if not equivalent to, the concepts related to ERP.

In our approach, the names of the concepts are compared as well as the relationship type. We use the n-grams [Damashek, 1995] algorithm to compare the labels of the related concepts and the relationships, i.e. Physician and instanceOf respectively. Using a string comparison that returns a decimal between 0 and 1, we recognize the related concepts as equivalent if their score is above

.70. Once all related concepts have been examined, the service and client concepts are believed to be equivalent if the weighted average score is .70 or greater. The weighted average score for a concept is shown below. This approach, although using syntactic comparison, uses semantics from the schema as well.

$$\frac{\textit{(totalFromComparisonOfEachRelatedConcept)}}{\textit{(totalNumberOfRelationshipsFromClientConcept)}} \, *.70$$

Figure 15 - Computing Weighted Average

Another approach that we considered was to create SWRL (Semantic Web Rule Language) rules that were specific enough to capture relationship information regarding a resource but general enough to be applicable to an entire class. For example, we could have created a rule that a technician, like ‘Radiology Tech, belonging to the same department, ‘Radiology.’, as another technician, ‘Radiology Technician, are equivalent concepts as seen in Figure 16. However, this approach was forgone due for two reasons. After analysis of the data in the ontologies, there would be an unacceptable of false positives. Secondly in a domain less structure than the HL7 domain, it would require many rules a priori knowledge of the ontology.

```
Technician(_department) → sameAs(Technician, _department)
```

Figure 16 - SWRL Technician Rule

A caveat to our approach is the Session Role, as mentioned in section 3.5.2. A Session Role is a functional requirement in which while a user is performing the functions of a given role, it may not perform the functions of some other role(s). In this scenario, it is necessary to implement rules and query of a rule engine, such as those provided with Jena.

2.6.3.1 Ontology Based Inferencing

Inference engines, also called reasoners, are software applications that derive new facts or associations from existing information. Inference and inference rules allow for deriving new data from data that is already known. Thus, new pieces of knowledge can be added based on previous ones. By creating a model of the information and relationships, we enable reasoners to draw logical conclusions based on the model. For example, with OWL it is possible to make inferences based on the associations represented in the models, which primarily means inferring transitive relationships. Jena has a built in rule-base reasoner that provides OWL inferencing support. The RBAC standard requires the ontology have the ability to apply multiple restrictions and cardinalities to concepts in the hierarchy. The reasoner that is built into Jena is able to reason over these more complex ontologies. Let us see a concrete example of how authorization can use inference mechanisms. The following example, which we discussed in section 2.2, is from our HL7 ontology and will be used in the evaluation of our approach.

In Figure 17 below, the circles are concepts and the diamonds are instances from the HL7 RBAC ontology. The figure depicts an ‘Emergency Room Physician’ who works for ‘St. Francis Hospital’. Referring to our running scenario, suppose that one criterion for accessing a patients’ EHR (Electronic Health Record) is as follows: must be a ‘Physician’ and must work in ‘Health Services’. Through the use of the HL7 RBAC ontology and domain knowledge we can infer that an ‘Emergency Room Physician’ whom works for ‘St. Francis’ does in fact meet the above criteria.

An ontology is a directed graph. Therefore, since in our example we begin with Physician, which is a criterion, it must be possible to detect that an 'Emergency Room Physician' is an 'instance of' 'Physician' in order to accurately make our prediction. Furthermore, we must be able to infer that a 'Hospital' is a 'Health Care Organization' which 'provides' 'Health Services'; as well as detect 'St. Francis' is an 'instance of' 'Hospital'. These can be seen within the perforated lines in the Figure 16.

Without inferencing, traversing the directed graph will not produce an accurate result, i.e. we would not be able to determine the relationship between 'Health Services' and 'Hospital'. The Jena API [Jena, 2006] provides multiple reasoners over various ontology languages. In our implementation we use the OWL full reasoner with SPARQL (SPARQL And Protocol RDF Query Language).

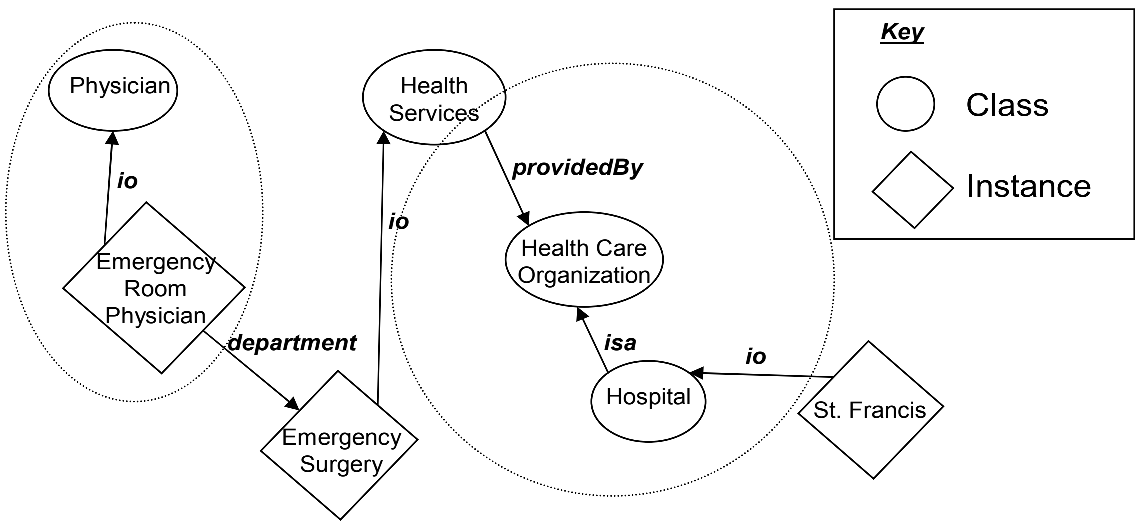


Figure 17 - Directed Graph of relationships

The example above requires two conditions are met. We break this down into two separate queries. Should the first query not determine a path, then the second query is aborted. This provides quicker results, especially when there are many conditions to be met on a large ontology. Below is an example SPARQL query.

```
SELECT ?x
WHERE { <Health Services> ?x <St. Francis> }
```

Figure 18 – SPARQL Query Example

The query in Figure 18 will not determine the path between 'Health Services' and 'St. Francis'. Therefore, we expand the query but adding variables, Figure 19. Our system has a default of five (5) expansions to the original query; however this can be increased or decreased per the clients' requirements. Because of the structure of our ontology, the necessary concepts to predict authorization are within our default value.

```
SELECT ?x
WHERE { <Health Services> ?x <St. Francis> }


SELECT ?x
WHERE { <Health Services> ?x ?y} {?y ?w <St. Francis> }


SELECT ?x
WHERE { <Health Services> ?x ?y} {?y ?w ?z} {?z ?k <St. Francis> }
```

Figure 19 - Query Expansion

It is becoming possible to perform these queries without the use of a 'brute force' method such as manually expanding our queries. [Anyanwu, Sheth, 2003] discusses the use of a ρ operator for queries that generalizes association based on structural properties and then queries based on these associations. This approach represents an ontology as a graph exploiting the relationships between concepts, therefore queries regarding these relationships, like the queries we are performing, can be reduced to graph path searches. This provides greater efficiency than dynamically expanding a query.

2.6.3.2 BPEL Process Policy: ClientDescription

In this section we will discuss the client authorization information which is placed inside the BPEL process policy. It uses the same local namespace as that of the annotations, i.e. ‘wsau’. The name which follows the local name is ClientDescription. This signals to the Authorization Predication manager that this is client side information.

After the local namespace and local name there are a series of attributes called ‘name#’, where ‘#’ is replaced by a number. These attributes are parsed and stored into a vector which is later used for ontology inferencing along with the concepts from the server side annotations. An example of the ClientDescription is depicted in Figure 20.

```
<wsp:Policy
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:base="http:/http://www.stfrenacishospital.com/policies"
xmlns:wsrm="http://schema.xmlsoap.org/ws/2004/03/rm"
xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau"
xmlns:Ontology1="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/ontologies/HL7_RBAC.owl">
 <wsp:ExactlyOne>
  <All>
    <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital" name3="St. Francis" />
     <sp:Basic256Rsa15 />
  </All>
  <All>
     <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital" name3="St. Francis" />
     <sp:TripleDesRsa15 />
  </All>
   <All>
    <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital" name3="St. Francis" />
  </All>

</wsp:Policy>
```

Figure 20 -BPEL Process Policy Files

A caveat to our approach is the case in which there is only one service a client, the requester, wishes to discover. In this case it is unlikely that the client will create a BPEL process. Using METEOR-S the client can create a Semantic Template and associate a client policy. Since prior to constraint analysis we construct a client policy for each task the client authorization information may be included in the client policy that is associated with the Semantic Template.

2.6.4 Evaluation of Technique

As we stated in section 2, we are focusing on three aspects of the HL7 scenario. The first service is 'Patient Identification and Lookup'. The second and third services are 'Review Medical History' and 'Review Problem Lists'. We believe that this is adequate to demonstrate and evaluate our approach. In our implementation it is assumed that 'Patient Identification and Lookup' must be preformed at least once prior to the second or third operations. We have added semantic annotations to the WSDL files for semantic discovery, as describe in [Akkiraju et al., 2005] [Verma et al., 2004]. In addition, we have attached our annotated policy files to the WSDL-S files.

We have created three fictitious companies to act as Electronic Health Record repositories: Company1, Company2, and Company3. Each repository has a Web service for each of the operations from the scenario. Therefore, there are nine total operations implemented, three operations for each of the three repositories. We have not named the implemented services the same as in the HL7 specification. This provides for a more real world evaluation.

In our evaluation we have tested for predictability of authorization during the semantic discovery phase of Web services. The Web service policies for Company1 and Company2 were annotated with authorization information for granting of authorization and denial of authorization, respectively. Company3 had been annotated with a combination of granting and denial information. The third case tests our approach of selecting an entire suite of Web services to satisfy a BPEL process.

The first test we conducted was an ‘instance of’ scenario, which has been our running example throughout the paper. The *Client* is an ‘Emergency Room Physician and works for St. Francis Hospital. The services of Company1 had been annotated such that authorization is to be granted to a client that has the *role* ‘Physician’ working in ‘Health Services’, *subjectCategory*. The services of Company2 had been annotated such that authorization is to be granted to a client that has the *role* ‘Nurse’ working in ‘Health Services’. The services of Company3 had authorization information which was a combination of the previous companies in which one service grants and two deny authorization. As we saw in section 5, St. Francis is a ‘Hospital’ which is an instance of an ‘Organization’ involved in ‘Health Services’. This test evaluated the multiple conditions present in the policy.

The Second test was a subsumption test, ‘Subclass of’. The *Client* is the ‘Chief of Medical Staff’ which is a ‘Subclass of’ ‘Executive Administration’ position, which is a ‘Subclass of ‘Administration’. The services of Company1 had been annotated such that authorization would be granted to a client that has the *role* ‘Administration’. The services of Company2 had been annotated such that authorization would be granted to a ‘General Employee’. The services of Company3 had authorization information which was a combination of the previous companies in which two services would grant and one denies authorization.

The third test we conducted was an equivalence test. The *Client* is a ‘Physician’. The services of Company1 had been annotated such that authorization would be granted to a client that has the *role* ‘Physician’. The services of Company2 had been annotated such that authorization is to be granted to a client that has the *role* ‘Nurse’. The services of Company3 had authorization information which was a combination of the previous companies in which two services would grant and one denies authorization

The fourth test was structured to evaluate the relationship between *permissions* and *role*s. The *Client* is again an ‘Emergency Room Physician’. The services of Company1 had been annotated with the appropriate *permissions* assigned to each service. For instance, ‘Patient Identification and Lookup’ is annotated to a lookup service, ‘Review Medical History’ is annotated to a medical history service, and ‘Review Problem Lists’ is annotated to a problem list service. The services of Company2 had been have *permissions* which the ‘Emergency Room Physician’ does not have assigned to each of the three operations. The services of Company3 have been annotated with *permissions* from the first triad and one service annotated as in the second triad.

The final test we preformed evaluates combining a service from each company in order to create a BPEL process. In this test we have reused our fourth test altering the authorization information. The first service from Company1, the second service from Company2, and the third service from Company3 had been annotated such that the client should predict that authorization will be granted. All other services, the client should predict that authorization will be denied.

In the above scenarios we were able to determine all paths between the client information and the concepts used to annotate the policy file. This proved that if the concepts are both found in one ontology and there is an authorization relationship, since it is an RBAC ontology, then our system will determine it. We further tested discrepancies between the ontologies, as we discussed in section 2.5 and 2.6.3. The focus of this testing was on the concept comparison which we perform if an authorization relationship does not exist due to synonymous or unrelated concepts. Table 3 shows the concepts from each ontology which were used to annotated the policy files for our testing, the number of relationships the client concept has and the number of similar relationships the concepts have with each other.

CLIENT:	**SERVICE:**	**# CLIENT:**	**#COMMON:**	**DETERMINE EQUIVALENCY**
MRI_ Technician	Magnetic_Resonance_ Investigations_Tech	5	3	**Yes**
Radiology_ Technician	Med_Tech	3	1	**No**
Nuclear_Med_ Technician	Radiology_Nuc_ Med_Tech	4	2	**Yes**
Emergency_ Physician	Emergency_Room_ Physician	4	3	**Yes**
Interventional_ Physician	Interventional_ Radiologist	5	1	**No**
CC_RN	ICU_RN	3	1	**No**
Nurse_Anesthetist	Nurse_Surgical	3	2	**Yes**
Pediactic_Nurse	Nurse_Pediatric	3	2	**Yes**

Figure 3 - Concept Relationships

The last column displays whether the client concept was found to be equivalent to the service concept. In most cases when it was equivalent this was accurately determined. Let us focus on some of the more interesting results. First, the "Interventional Physicians' are equivalent to 'Interventional Radiologist'. This equivalency was not determined because of the lack of common relationships, false negative. We did purposely reduce the number of common relationships for our test. However, we did keep with what could potentially happen in a real world scenario, that is one organization may have a more robust Lower Level Ontology than another organization. Another interesting result is the determination of equivalency of a 'Anesthetist' and 'Surgical' Nurse. These are indeed two different concepts; however, both perform their duties in a surgical room and both are instances of Nurse. With only one other relationship to differentiate the two, we encountered this false positive.

Overall, incorrect predictions were approximately 20% of our total predication testing. When conducting our discrepancy testing, these concepts did pass through our authorization relationship algorithm as well. Since no relationship was found, it was then passed to our concept comparison algorithm. We feel that these results are positive and that this is good indication of authorization predication.

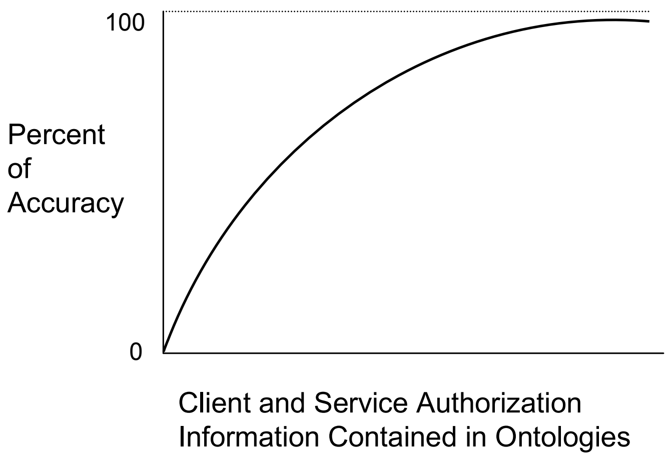


Figure 21 - Information vs. Accuracy

Figure above shows the correlation between the amount of information stored in an ontology and the percentage of accuracy. As the amount of information increases, so does the accuracy of our prediction. Our prediction is directly effected by the number and types of relationships found in the Lower Level Ontologies used to express authorization requirements as well as the clients role. For domains that have not been standardized as the HL7 domain has, we would expect a lower percentage of accuracy. However, it is our opinion that the accuracy would increase as more information is placed into the ontologies.

2.7 Related Work

Authorization in Web services is currently a research area of great interest. WS-Authorization is one of a few Web service specifications remaining to be standardized [IBM, Microsoft 2002]. In the Semantic realm, including both Semantic Web and Semantic Web services, research interests are rapidly growing.

Some previous research regarding authorization in Semantic Web services has been focused on implementing an access control enforcement structure [Yague et al., 2003]. [López et al., 2005] discusses an approach to access control in a distributed heterogeneous network in which XACML and SAML are used for access control. The novelty of their approach is to convert between these standards for the enforcement of policies.

[Agarwal et al., 2004] uses attributes from credentials like SAML or Digital Certificates to make access control decisions in their implementation. While this can be done, these credentials were designed for authentication. Our approach is similar to [Kagal et al. 2004], in which the authors use ontologies to add authorization annotations to OWL-S. OWL-S is used to add semantics to Web services. This is done through a mapping of concepts in OWL-S to WSDL types. Their approach adds extensibility elements to the OWL-S constructs for the purpose of supplying authorization information. As mentioned in section 6.2, our approach is from the WSDL-S perspective.

An idea that has gained momentum recently is that of a hybrid approach. This approach incorporates real world concepts in ontology with rule based ontology and is described in [Kagal et al. 2004]. The strength of their approach is in describing access control policies with multiple ontologies. This provides for greater expressiveness since the semantics of rules may be incorporated.

Our work differs from the previous works in several ways. We are extending the accepted standard WSP by adding semantic annotations. This is different because instead of creating an entirely new standard we are building on an accepted standard. In addition, we are using the RBAC standard in our annotation scheme and for our ontologies. Other approaches have developed their own authorization ontologies. Furthermore, other approaches have named their extensibility elements as they see fit; not taking into account accepted standards. Our evolutionary approach builds on current standards where these other approaches are more revolutionary.

Policy matching in Semantic Web services is a complicated area of research. [Verma et al., 2005] details an implementation of Semantic Policy matching using the Semantic Web Rule Language (SWRL). Our Authorization Prediction is a module that can be plugged into METEOR-S' constraint analysis.

[Wu et al., 2002] describes how to incorporate access control in a business process (workflow). It illustrates fundamental capabilities in a workflow and why authorization and access control need to be expressed semantically. This is particularly relevant to our research on the client side where we had to decide on the appropriate location for client information.

[Anyanwu et al., 2003] examines the issues of complex processes inherent in health care applications in a heterogeneous cross domain environment. The authors suggest the use of workflows to coordinate the health care applications' processes.

2.8 Conclusion and Future Work

Many researchers believe that a new Web will emerge in the next few years based on the large-scale research and developments ongoing on the semantic Web and Web services. Nevertheless, the industry and its main players are taking a “wait-and-see” approach to see how real-world applications can benefit from semantic Web technologies. The success of the semantic Web and semantic Web services vision is dependant on the development of practical and useful semantic Web-based applications.

While the semantic Web has reached a considerable stability from the technological point of view with the development of languages to represent knowledge (such as OWL), to query knowledge bases (SPARQL), and to describe business rules (such as SWRL), the industry is still skeptic on its potential. For the semantic Web to gain a considerable acceptance from the industry it is indispensable to develop real-world semantic Web-based applications to validate and explore the full potential of the semantic Web.

In their recent book, "Semantic Web Services, Processes and Applications”, Cardoso and Sheth clearly shown that the success of the semantic Web depends on its capability to support applications in commercial settings.

To take the development of semantic Web applications a step further, we have proposed a framework for the annotation of Web service policy files to incorporate authorization information in an effort to aid in the semantic discovery of authorized services. So far researchers have concentrated their attention on annotating Web services’ inputs, outputs, pre-conditions and post-conditions with semantics. While the effort improves the efficiency and precision of Web service discovery, it may not be sufficiently useful if clients or service requesters are not authorized to invoke the retrieved services. As an important addition to previous work, we extend accepted policy standards by adding semantic annotations. The use of standards is indispensable to guarantee a future adoption of new technologies by the industry.

As this paper examines authorization in Semantic Web services, we looked at the specification for securing Web services and found authorization was yet to have a specification. Because of RBAC's distributed nature and the XML representation of RBAC, XACML, it is a logical evolutionary step for expressing authorization semantics. We have proposed a framework for the annotation of Web service policy files to incorporate authorization information in an effort to aid semantic discovery of Web services.

Our approach contributes to this area of research in the following ways. Our extensible elements provide authorization criteria information regarding a Web service to a potential client without divulging any sensitive security information. The placement of client information into BPEL's WSDL process policy file, which provides an efficient and reusable method for making this information available for authorization prediction. The development of the RBAC Upper Level Ontology and HL7 RBAC Mid Level Ontology, which have been made available to the public on the LSDIS web site. Another contribution is the function definition of authorization for Web services and the comparison of that to our prediction function. This function clarifies the discrepancies which can affect the accuracy of authorization predication. The creation of a flexible XSD for WS-Authorization which allows semantic information to be contained within the elements.

The evaluation of our implementation shows the usefulness of our approach. We believe that this approach can be applied to other domains which have equivalent standards. For instance, in ecommerce there is the RosettaNet standard for transactions between business partners. With an appropriate RBAC ontology and RosettaNet ontology, our approach can be applied successfully.

Future work should be focused on formalizing the WS-Authorization framework. Included in this should be a formal description for adding authorization semantics to WS-Policy. Our approach is the first step toward semantic access control. Future work would include a semantic policy decision point and semantic policy enforcement point.

Additional future work in this area is the creation of multiple lower level ontologies which can be imported into the HL7 ontology for further testing. Also, the creation of ontologies for other domains built on the RBAC upper level ontology. Lower level ontologies for these domains are also necessary to test our approach because most domains are not structured and defined as clearly as the Health Care domain.

Results will vary from domain to domain and ontology to ontology. Our results show that a client can increase its confidence in Web services discovery through our process. As far as we know, our approach is the first of its kind and is a small step in the overall goal of semantic security. However, our results also demonstrate that the more information in common to the client and service provider the more accurate the prediction; which is a good argument for deeply structured upper level ontologies.

Our positive results were achieved through our dedication to the use of accepted standards and the structure of the domain in which we were applying our approach. HL7 is the standard for the health care domain provided most role names and the permission sets. The RBAC standard is a highly accepted standard for access control, and use of this standard in our extensibility elements provided us with concrete descriptions for our annotations. Our ontology combined these two standards when we used the HL7 requirements for RBAC.

ABOUT THE AUTHORS

Richard S. Patterson is a Masters candidate of Computer Science at the University of Georgia. His research interests include Web services, Semantic Web, Security, and Access Control. Richard received his B.B.A. in Economics form the University of Georgia in 1998. Richard worked for five years in IT architecture and Security Consulting before returning to pursue a Masters degree.

John A. Miller is a Professor of Computer Science at the University of Georgia and has also been the Graduate Coordinator for the department for 9 years. His research interests include database systems, simulation, bioinformatics and Web services. Dr. Miller received the B.S. degree in Applied Mathematics from Northwestern University in 1980 and the M.S. and Ph.D. in Information and Computer Science from the Georgia Institute of Technology in 1982 and 1986, respectively. During his undergraduate education, he worked as a programmer at the Princeton Plasma Physics Laboratory. Dr. Miller is the author of over 100 technical papers in the areas of database, simulation, bioinformatics and Web services. He is an Associate Editor for *ACM Transactions on Modeling and Computer Simulation* and *IEEE Transactions on Systems, Man and Cybernetics* as well as a Guest Editor for the *International Journal in Computer Simulation* and *IEEE Potentials*.

Jorge Cardoso joined the University of Madeira (Portugal) in March 2003. He previously gave lectures at University of Georgia (USA) and at the Instituto Politécnico de Leiria (Portugal). Dr. Cardoso received his Ph.D. in Computer Science from the University of Georgia in 2002 (with Amit Sheth). While at the University of Georgia, he was part of the LSDIS Lab, where he did extensive research on workflow management systems. In 1999, he worked at the Boeing Company on enterprise application integration with Christoph Bussler. Dr. Cardoso was the co-organizer and co-chair of the First, Second, and Third International Workshop on Semantic and

Dynamic Web Processes. He has published over 55 refereed papers in the areas of workflow management systems, semantic Web, and related fields. He has also edited 3 books on semantic Web and Web services. Prior to joining the University of Georgia, he worked for two years at CCG, Zentrum für Graphische Datenverarbeitung, where he did research on Computer Supported Cooperative Work.

Mike Davis VHA Security Architect, CISSP Mr. Davis is a Senior Security Engineer with Science Applications International Corporation (SAIC) with over seven years experience in providing security solutions to the healthcare industry, five of those as Security Architect for the Veterans Health Administration, Office of the CIO, Health Information Architecture Office. He currently develops VHA architectures for PKI, identity management and electronic signature. He manages VHA's ESIG lab and is piloting Veteran ESIG solutions at VHA facilities in support of VHA's more than 7 million Veterans. He represents the VHA on several Standards Development Organizations including ASTM, HL7 and OASIS developing healthcare industry standards for electronic signature, PKI, RBAC and identity management.

Chapter 3

CONCLUSION

In this work we discussed the currently active research area of authorization in a Semantic Web services environment. We looked at the specification for securing Web services and found authorization missing. We then reviewed the common authorization techniques, finding only one of them suitable for Web services. It was because of RBACs' distributed nature and the XML representation of RBAC, XACML, which made it a logical evolutionary step for expressing authorization semantics.

There are several contributions in this research. There are the extensible elements which are the foundation for the annotations. There is the use of currently accepted and implemented standards for these extensible elements, RBAC and XACML. Another benefit is that authorization moves from the application which the Web service invokes, like a database, to the Web service. This provides an additional layer of indirection for security. In addition, the novel concept of a Client Authorization Cache provides additional information for predicting authorization.

Future work in this area includes the development of ontologies for each of the industry sectors, further development of the constraint analysis, and a protocol for passing CAC information between requester and provider.

Chapter 4

REFERENCES

[ActiveBPEL 2.0, 2006] Copyright © 2004–2006, ActiveBPEL, LLC., http://www.activebpel.org/

[Active Enpoints, 2006] Copyright © 2004-2006 Active Endpoints, http://www.active-endpoints.com/products/activebpeldes/index.html

[Akkiraju et al., 2005] Akkiraju R, Farell J, Miller J, Nagarajan M, Sheth A and Verma K, "Web Service Semantics - WSDL-S," Proceedings of the W3C Workshop on Frameworks for Semantics in Web Service (W3CW'05), Innsbruck, Austria (June 2005) pages 5.

[Aleman-Menza et al., 2006] Boanerges Aleman-Meza, Meenakshi Nagarajan1, Cartic Ramakrishnan1, Li Ding, Pranam Kolari, Amit P. Sheth1, I. Budak Arpinar, Anupam Joshi, Tim Finin, International World Wide Web Conference, Proceedings of the 15th international conference on World Wide Web Edinburgh, Scotland, SESSION: Social networks, 2006, pp 407 - 416

[Anyanwu et al., 2003] Kemafor Anyanwu, Amit P. Sheth, Jorge Cardoso, John A. Miller and Krys J. Kochut, "Healthcare Enterprise Process Development and Integration," *Journal of Research and Practice in Information Technology (JRPIT),* Special Issue on Health Knowledge Management, Vol. 35, No. 2 (May 2003) pp. 83-98. Australian Computer Society, Inc.

[Anyanwu, Sheth, 2003] Kemafor Anyanwu and Amit P. Sheth, “The ρ Operator: Discovering and Ranking Associations on the Semantic Web” ACM SPECIAL ISSUE: Special section on semantic web and data management, Volume 31 , Issue 4 2002 pp 42 - 47

[Agarwal et al., 2004] S. Agarwal, B. Sprick, S. Wortmann; Credential Based Access Control for Semantic Web Services; http://www.aifb.uni-karlsruhe.de/WBS/sag/papers/Agarwal_Sprick_Wortmann-CredentialBasedAccessControlForSemanticWebServices-AAAI_SS_SWS-04.pdf.

[Census 2000] Census Bureau, 2000 Industry Categories for the Special EEO File, 2000.

[Christensen et al. 2001] Christensen E., Curbera F., Meredith G. and Weerawarana S., 2001, Web Services Description Language (WSDL) 1.1, W3C Note, http://www.w3.org/TR/wsdl.

[Damashek1 1995] M. Damashek. *Gauging similarity with n-grams: language independent categorization of text*. Science, 267(5199) pp 843--848, 1995

[Dogac et al., 2002] Dogac A., Cingil I., Laleci G., Kabak Y., Improving the Functionality of UDDI Registries through Web Service Semantics, 3rd VLDB Workshop on Technologies for Eservices (TES-02), Hong Kong, China, August 23-24, 2002

[Dong et al., 2005] X.L. Dong, A. Halevy and J. Madhavan (2005) *Reference reconciliation in complex information space*. In Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, ACM Press: Baltimore, MD. Pp. 85-96

[FaCT 2005] FaCT (2005) FaCT++, http://owl.man.ac.uk/factplusplus/.

[Fensel et al.] Fensel D. and Bussler C., The Web Service Modeling Framework WSMF, http://informatik.uibk.ac.at/users/c70385/wese/wsmf.paper.pdf

[Gavirla et al.] S. Gavrila, D. Kuhn, R. Chandramouli; *Proposed NIST Standard for Role-Based Access Control*; http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf

[Gandon and Sadeh 2003] Gandon, F. L. and N. M. Sadeh, OWL inference engine using XSLT and JESS, http://www-2.cs.cmu.edu/~sadeh/MyCampusMirror/OWLEngine.html, 2003.

[HL7] HL7 http://www.hl7.org

[HL7 Security Technical Committee, 2005] HL7 Security Technical Committee, Role Based Access Control (RBAC) Healthcare Scenarios Version 1.0, 2005.

[HL7 Security Technical Committee, 2005] HL7 Security Technical Committee, Role Based Access Control (RBAC) Healthcare Permissions Catalog Version 2.0, 2005

[Hull and East, 2006] Hull and East Yorkshire Hospitals NHS Trust, 2006.

[IBM, Microsoft 2002] IBM Corporation and Microsoft Corporation, Security in a Web Services World: A Proposed Architecture and Roadmap Version 1.0, 2002.

[Jena, 2006] Hewlett-Packard Development Company, LP., 2006. http://jena.sourceforge.net/

[Kagal et al. 2004] L. Kagal, M. Paolucci, N. Srinivasan, G. Denker, T. Finin, K. Sycara; Authorization and Privacy for Semantic Web Services; IEEE Intelligent Systems (Special Issue on Semantic Web Services), July 2004.

[Leymann et al. 2002] Leymann F, Roller D, Schmidt MT - IBM Systems Journal, 2002 Web services and business process management

[López et al., 2005] G. López, Ó. Cánovas, A. Gómez-Skarmeta, S. Otenko, D. Chadwick; A Heterogeneous Network Access Service based on PERMIS and SAML; In Proceedings of 2nd EuroPKI Workshop, University of Kent, July 2005.

[NIST, 1993] National Institute of Standards and Technology (NIST) FIPS Publication 180: Secure Hash Standard (SHS). May 1993.

[NIST, 2004] National Institute of Standards and Technology (NIST) Role Based Access Control Standard (RBACS). April 2004.

[OWL, 2004] Deborah L. McGuinness, Frank van Harmelen, W3C Recommendation 10 February 2004

[Pellet, 2003] Minswap, http://www.mindswap.org/2003/pellet/ , 2003

[Pennington, 2006] Cary Pennington, "Policy Based Optimal Composition of Web Services," Masters Thesis (M.S. in CS Degree) July 2006.

[Rosenberg, Remy, 2004] Jothy Rosenberg and David Remy, *Securing Web Services Security with WS-Security*, Sams, 2004.

[SAML 2.0, 2005] SAML 2.0 profile of XACML v2.0 OASIS Standard, 1 February 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf

[SAWSDL, 2006] Semantic Annotations for WSDL Editors Copy, Joel Farrell, IBM
Holger Lausen, DERI Innsbruck, August 08, 2006.
http://www.w3.org/2002/ws/sawsdl/spec/SAWSDL.html

[Sirin et al., 2004] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. Submitted for publication to Journal of Web Semantics.

[Sivashanmugam et al. 2003] Sivashanmugam, K., Verma, K., Sheth, A., Miller, J., Adding Semantics to Web Services Standards, Proceedings of the 1st International Conference on Web Services (ICWS'03), Las Vegas, Nevada (June 2003).

[Toninelli et al., 2005] A. Toninelli, J. Bradshaw, L. Kagal, R. Montanari; Rule-based and Ontology-based Policies: Toward a Hybrid Approach to Control Agents in Pervasive Environments; Proceedings of the Semantic Web and Policy Workshop, International Semantic Web Conference, 7 November, 2005.

[UDDI, 2002] UDDI Spec Technical Committee Specification, 2002.
http://uddi.org/pubs/uddiv3.00- published-20020719.htm

[Verma et al., 2005] Verma K, Sivashanmugam K, Sheth A, Abhijit Patil, Oundhakar S and Miller J, METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services, Journal of Information Technology and Management, Special Issue on Universal Global Integration, Vol. 6, No. 1 (2005) pp. 17-39. Kluwer Academic Publishers.

[Verma, 2006] Kunal Verma, "Configuration and Adaptation of Semantic Web Processes," Doctoral Dissertation (Ph.D. in CS Degree) June 2006

[Verma et al., 2005] K. Verma, R. Akkiraju, R Goodwin; Semantic Matching of Web Service Policies; Second International Workshop on Semantic and Dynamic Web Processes (SDWP 2005), Third International Conference on Web Services (ICWS'05), July, 2005.

[Verma et al., 2004] Verma K, Aggarwal R, Miller J and Milnor W, "Constraint Driven Web Service Composition in METEOR-S," Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04), Shanghai, China (September 2004) pp. 23-32

[Wielemaker 2005] Wielemaker, J., SWI-Prolog Semantic Web Library, http://www.swi-prolog.org/packages/semweb.html, 2005.

[WS Architecture, 2004] Web Services Architecture (WS Architecture), D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, D. Orchard; http://www.w3.org/TR/ws-arch/#security Feb. 2004

[WS-Policy] *et al* Siddharth Bajaj, Don Box; *Web Services Policy Framework (WS-Policy)*; ftp://www6.software.ibm.com/software/developer/library/ws-policy.pdf

[WS-Security, 2002] Web Services Security (WS-Security) Version 1.0 05, 2002 *et al* Bob Atkinson, Giovanni Della-Libera; *Specification: Web Services Security)*; ftp://www6.software.ibm.com/software/developer/library/ws-secure.pdf; April 2002

[Wu et al., 2002] S. Wu, A. Sheth, J. Miller, Z Luo; Authorization and Access Control of Application Data in Workflow Systems; Journal of Intelligent Information Systems: Integrating Artificial Intelligence and Database Technologies (JIIS), Vol. 18, No. 1 (January 2002) pp. 71-94. Kluwer Academic Publishers.

[XACML, 2005] eXtensible Access Control Markup Language, (XACML) Version 2.0 OASIS Standard, 1 Feb 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[XML-Signature, 2002] XML Signature Syntax and Processing (XML-Signature) W3C Recommendation 2002 http://www.w3.org/TR/xmldsig-core/

[XML-Encryption, 202] XML Encryption Syntax and Processing (XML-Encryption) W3C Recommendation 2002 http://www.w3.org/TR/xmlenc-core/

[Yague et al., 2003] M. Yague, A. Mana, J. Lopez, J. Troya; Applying the Semantic Web Layers to Access Control; 14th International Workshop on Database and Expert Systems Applications (DEXA'03)

APPENDIX A – HL7 Scenario in Entirety

"Mr. Patient was placed in a clinic examination room for a Diabetic Consultation. The Physician greeted Mr. Patient and accessed Mr. Patient's medical records <<PRD-006 Patient Identification and Lookup>> from an EHR. After briefly reviewing Mr. Patient's recent pertinent medical history <<PRD-003 Review Medical History>>, problem lists <<PRD-016 Review Problem Lists>>, health status data <<PRD-014 Health Status Data>>, existing order(s) <<PRD-004 Review Existing Order(s)>>, medications <<PRD-010 Review Patient Medications>>, allergies <<PRD-011 Review Patient Allergies or Adverse Reactions>>, test results/reports (evaluating high and low values) <<PRD-001 Review Patient Testing Reports>>, immunizations <<PRD-013 Review Immunizations>>, and visits <<PRD-012 Review Past Visits>> within the EHR, the Physician next reviewed Mr. Patient's vital signs, patient measurements <<PRD-005 Review Vital Signs/Patient Measurements>> (e.g., height and weight), and the chief complaint(s) <<PRD-002 Review Chief Complaint>> that were entered for this encounter."

The Physician asked Mr. Patient about any problems or concerns. The Physician also asked Mr. Patient about compliance with diet, exercise, and medication regime for diabetes. Mr. Patient admitted that although he was fairly diligent with his exercise program, he had some problems maintaining his dietary regime. The Physician entered this information into the EHR as part of the subjective findings for this encounter.

The Physician also noted from reviewing today's encounter in the EHR that Mr. Patient had already been sent to the lab by the screening nurse to draw blood specimens for random blood glucose and a hemoglobin A1C for this appointment based on the Physician's pre-visit planning test orders. The orders were placed during the previous visit. The Physician also noted that the results from today's lab tests were not available yet.

The Physician did an examination of Mr. Patient, and entered the results of the examination into the EHR. In addition, the Physician accessed the Diabetes Clinical Guidelines <<PRD-007 Review Patient or Disease-Specific Clinical Guidelines>> within the EHR and compared them to Mr. Patient's current treatment status and plan. Upon his earlier review of the EHR, he noted that Mr. Patient had not had an electrical cardiogram (ECG) recently, so he ordered an ECG for Mr. Patient via the EHR and sent him to the Cardiology Department for the study.

Upon Mr. Patient's returning from getting the ECG, he was placed in the Physician's examination room. When the Physician entered the exam room, he accessed Mr. Patient's records in the EHR and looked at the ECG that was just done, which showed a normal sinus rhythm with no acute changes since the previous ECG performed 7 years ago. The Physician then accessed Mr. Patient's lab results, which were now available, and found that the random blood glucose was 165 and the hemoglobin A1C was 6.8. The Physician also reviewed previous results for Mr. Patient's hemoglobin A1C and blood glucose for comparison.

The Physician discussed these results with Mr. Patient and explained that the blood tests indicated that his diabetes was not currently under tight enough control. They both agreed to a trial at better compliance with a diabetic dietary and exercise program before a change in medication. The Physician accessed the Current Directory of Provider Information <<PRD-009 Review Current Directory of Provider Information>> in the EHR to locate telephone number for the Registered Dietitian to ask some questions about the current status of recommending low glycemic-indexed foods for diabetes and also entered an order for a dietary consult for Mr. Patient in the EHR, requested for the following week.

The Physician then ordered a new prescription for metformin and a refill of glyburide at the same doses that Mr. Patient was previously taking, viewing prescription costing information of generic brands <<PRD-015 Review Prescription Costing Information>>. The following week, the Physician received an alert <<PRD-008 Review Alerts>> from the EHR that the dietary consult for Mr. Patient was complete. The Physician accessed Mr. Patient's dietary consult and reviewed the findings." [HL7 Security Technical Committee, 2005]

APPENDIX B – Example Details

Example WSDL

```
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:ns1="http://AmericanHealthRecords.ElectronicHealthRecords/types"
xmlns:ns="http://www.AmericanHealthRecords.com"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
targetNamespace="http://www.AmericanHealthRecords.com">

<wsdl:types><xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:types="http://AmericanHealthRecords.ElectronicHealthRecords/types"
targetNamespace="http://AmericanHealthRecords.ElectronicHealthRecords/types"
elementFormDefault="unqualified" attributeFormDefault="unqualified">

<xs:element name="RetreivePatientid">
<xs:complexType>
 <xs:sequence>
  <xs:element type="xs:string" name="PatientName" />
  <xs:element type="xs:string" name="SSN" />
  <xs:element type="xs:string" name="PhysicianName" />
  <xs:element type="xs:string" name="password" />
 </xs:sequence>
</xs:complexType>
</xs:element>
```

```
<xs:element name="RetreivePatientidResponse">
<xs:complexType>
 <xs:sequence>
  <xs:element type="xs:string" name="return" />
 </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="MedicalHistory">
<xs:complexType>
 <xs:sequence>
  <xs:element type="xs:string" name="PID" />
  <xs:element type="xs:string" name="PhysicianName" />
  <xs:element type="xs:string" name="password" />
 </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="MedicalHistoryResponse">
<xs:complexType>
<xs:sequence>
 <xs:element type="xs:string" name="return" />
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="PatientProblemList">
<xs:complexType>
 <xs:sequence>
```

```
  <xs:element type="xs:string" name="PID" />
  <xs:element type="xs:string" name="PhysicianName" />
  <xs:element type="xs:string" name="password" />
 </xs:sequence>
</xs:complexType>
</xs:element>
 <xs:element name="PatientProblemListResponse">
<xs:complexType>
<xs:sequence>
 <xs:element type="xs:string" name="return" />
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="RetreivePatientidMessage">
  <wsdl:part element="ns1:RetreivePatientid" name="part1" />
</wsdl:message>
<wsdl:message name="RetreivePatientidResponseMessage">
  <wsdl:part element="ns1:RetreivePatientidResponse" name="part1" />
</wsdl:message><wsdl:message name="MedicalHistoryMessage">
  <wsdl:part element="ns1:MedicalHistory" name="part1" />
</wsdl:message>
  <wsdl:message name="MedicalHistoryResponseMessage">
<wsdl:part element="ns1:MedicalHistoryResponse" name="part1" />
</wsdl:message>
```

```
  <wsdl:message name="PatientProblemListMessage">
<wsdl:part element="ns1:PatientProblemList" name="part1" />
</wsdl:message>
  <wsdl:message name="PatientProblemListResponseMessage">
<wsdl:part element="ns1:PatientProblemListResponse" name="part1" />
</wsdl:message>
<wsdl:portType name="AmericanHealthRecordsPortType">
 <wsdl:operation name="RetreivePatientid">
  <wsdl:input message="ns:RetreivePatientidMessage" />
  <wsdl:output message="ns:RetreivePatientidResponseMessage" />
</wsdl:operation>
 <wsdl:operation name="MedicalHistory">
  <wsdl:input message="ns:MedicalHistoryMessage" />
  <wsdl:output message="ns:MedicalHistoryResponseMessage" />
</wsdl:operation>
 <wsdl:operation name="PatientProblemList">
  <wsdl:input message="ns:PatientProblemListMessage" />
  <wsdl:output message="ns:PatientProblemListResponseMessage" />
</wsdl:operation>
</wsdl:portType>
<wsdl:binding type="ns:AmericanHealthRecordsPortType"
name="AmericanHealthRecordsSOAP11Binding">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="RetreivePatientid">
<soap:operation style="document" soapAction="urn:RetreivePatientid" />
<wsdl:input>
```

```
<soap:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output><soap:body namespace="http://www.AmericanHealthRecords.com" use="literal"
/>
</wsdl:output>
</wsdl:operation><wsdl:operation name="MedicalHistory">
<soap:operation style="document" soapAction="urn:MedicalHistory" />
<wsdl:input>
<soap:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="PatientProblemList">
<soap:operation style="document" soapAction="urn:PatientProblemList" />
<wsdl:input>
<soap:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding type="ns:AmericanHealthRecordsPortType"
name="AmericanHealthRecordsSOAP12Binding">
```

```
<soap12:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="RetreivePatientid">
<soap12:operation style="document" soapAction="urn:RetreivePatientid" />
<wsdl:input>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="MedicalHistory">
<soap12:operation style="document" soapAction="urn:MedicalHistory" />
<wsdl:input>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="PatientProblemList">
<soap12:operation style="document" soapAction="urn:PatientProblemList" />
<wsdl:input>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body namespace="http://www.AmericanHealthRecords.com" use="literal" />
```

```
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="AmericanHealthRecords">
<wsdl:port binding="ns:AmericanHealthRecordsSOAP11Binding"
name="AmericanHealthRecordsSOAP11port">
<soap:address location="http://localhost:8080/axis2/services/AmericanHealthRecords" />
</wsdl:port>
<wsdl:port binding="ns:AmericanHealthRecordsSOAP12Binding"
name="AmericanHealthRecordsSOAP12port">
<soap12:address location="http://localhost:8080/axis2/services/AmericanHealthRecords" />
</wsdl:port></wsdl:service>
</wsdl:definitions>
```

Example Policy

```
<wsp:Policy
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
wsu:id="RM"
xmlns:base="http:/http://www.NationalEHR.com/policies"
xmlns:wsrm="http://schema.xmlsoap.org/ws/2004/03/rm"
xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau"
xmlns:Ontology1="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-
s/ontologies/HL7_RBAC.owl">
  <wsp:ExactlyOne>
```

```
   <wsp:All>
   <wsau:role name=”Emergency_Room_Physician"     wsau:ModelReference="Ontology1#
Emergency_Room_Physician "/>
   <wsau:subjectCategory name="Health_Services"
wsau:ModelReference="Ontology1#Health_Services"/>
   <wsse:SecurityToken>
          <wsse:TokenType>
                    wsse:X509v3
          </wsse:TokenType>
   </wsse:SecurityToken>
   </wsp:All>
  <wsp:All>
   <wsau:permission name=”read" wsau:ModelReference="Ontology1#read"/>
   <wsau:role name="Executive_Administration"
wsau:ModelReference="Ontology1#Executive_Administration"/>
   <wsau:subjectCategory name="Health_Services"
wsau:ModelReference="Ontology1#Health_Services"/>
  <wsse:SecurityToken>
          <wsse:TokenType>
                    wsse:X509v3
          </wsse:TokenType>
  </wsse:SecurityToken>
  </wsp:All>
</wsp:Policy>
```

Example Process WSDL policy

```
<wsp:Policy
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:base="http:/http://www.stfrenacishospital.com/policies"
xmlns:wsrm="http://schema.xmlsoap.org/ws/2004/03/rm"
xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau"
xmlns:Ontology1="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-
s/ontologies/HL7_RBAC.owl">
 <wsp:ExactlyOne>
  <All>
        <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital"
name3="St. Francis" />
  <sp:Basic256Rsa15 />
  </All>
  <All>
    <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital"
name3="St. Francis" />
    <sp:TripleDesRsa15 />
  </All>
   <All>
    <wsau:ClientDescription name1="Emergency Room Physician" name2="Hospital"
name3="St. Francis" />
  </All>
</wsp:Policy>
```

APPENDIX C – LDAP

LDAP provides some of the necessary relationships that are needed in a ubiquitous and dynamic environment through its Tree structure, Figure 1. Within the Tree there are containers which hold such things as countries, organizational units, people, resource, etc. When a directory entry is made, it associate lower lying containers, such as people or resources, with containers higher in the Tree, such as organizations (and visa versa). For example, there may be a relationship association as follows: a printer (resource) is located at the Riverview Branch (location) of ACME Bank (organization) which is in Atlanta Ga, (location) of the United Sates (country). In the same way, it is possible to find all printers within this directory that are located within the United States.

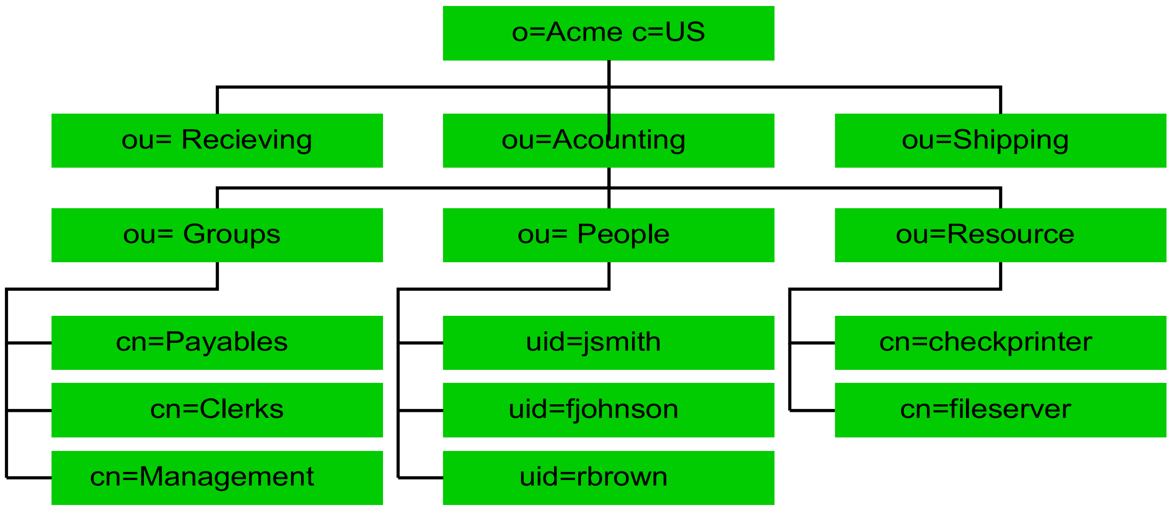


Figure 1 - LDAP Tree

LDAPs' access control uses ACL's. The benefit of using LDAP over a pure ACL implementation is that associations made between the containers within the directory allow for a more granular approach to managing the system. These relationships are put to use when access control is implemented. Resources can now be grouped together in a container which will need only one ACL. Furthermore the hierarchical structure of LDAP freely provides groups and sub-groups which can aid in managing ACL's.

Appendix D – WS-Authorization XSD

```
<?xml version="1.0" ?><schema xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 targetNamespace="http://lsdis.cs.uga.edu/authorization/wsau"
 xmlns:wsau="http://lsdis.cs.uga.edu/authorization/wsau">
  <element name="permission">
    <complexType>
      <complexContent>
            <restriction base="anyType">
              <attribute name="name" type="string" />
              <attribute name="modelReference" type="anyURI" use='optional'/>
              <attribute name="expression" type="string" use='optional'/>
            </restriction>
      </complexContent>
    </complexType>
  </element>
  <element name="role">
    <complexType>
      <complexContent>
            <restriction base="anyType">
              <attribute name="name" type="string" />
              <attribute name="modelReference" type="anyURI" use='optional'/>
              <attribute name="expression" type="string" use='optional'/>
            </restriction>
      </complexContent>
  </complexType>
 </element>
 <element name="subjectCategory">
  <complexType>
   <complexContent>
       <restriction base="anyType">
        <attribute name="name" type="string" />
        <attribute name="modelReference" type="anyURI" use='optional'/>
        <attribute name="expression" type="string" use='optional'/>
       </restriction>
   </complexContent>
  </complexType>
 </element>
  <element name="modelReference">
  <complexType>
   <complexContent>
       <restriction base="anyType">
        <attribute name="modelReference" type="anyURI" use='optional'/>
        <attribute name="expression" type="string" use='optional'/>
       </restriction>
   </complexContent>
  </complexType>
 </element>
</schema>
```