A Metadata-based Access Control Model for Web Services[*]

Mariemma I. Yagüe, Antonio Maña, Javier Lopez

Computer Science Department
University of Málaga. Spain.
{yague, amg, jlm}@lcc.uma.es

Abstract: One of the most relevant advantages of Web Services (WS) is their simplicity of access on the Internet. However, this feature also makes them vulnerable to a series of security threats. Additionally, the application of WS to many interesting problems is currently hindered by the lack of mechanisms that provide, among others, adequate access control functionalities for this scenario. In fact, access control and authorization are critical because of the specific characteristics of WS. When considering the requirements of this scenario we must highlight not only flexibility of the access control system for dissimilar security policies, but also the control over a large number of elements and the distributed nature of these ones. Other important issues are dynamism of the WS environment, and interoperability of authorization mechanisms for the integration of multiple WS from various sources. The present work introduces an access control model for WS that addresses all previous issues. The model is built on the basis of separation of the authorization and access control management responsibilities. We introduce mechanisms for the semantic integration of an external Privilege Management Infrastructure (PMI) and present the Semantic Policy Language (SPL) for the description of access criteria based on attribute certificates.

KeyWords: Web Services, Access Control, Authorization, Privilege Management Infrastructure, XML Metadata, Semantic Integration.

1 Introduction

Web Services (WS) are the basic building blocks in distributed computing on the Internet. Applications are built integrating multiple WS from various sources, which cooperate regardless of their location and implementation. Although we can find in the literature many definitions for "Web Services", most of them agree on the following:

- WS expose useful functionality to users through a standard Web protocol. In most cases, the protocol used is *Simple Object Access Protocol* (W3C, 2000).
- WS provide a way to describe their interfaces with enough detail as to allow a user to build a client application to access their functionality. This description is usually provided in an XML document called a *Web Services Description Language* document (W3C, 2001).
- WS are registered in a way that potential users can find them easily. This is done with *Universal Discovery Description and Integration* (OASIS, 2000).

[*] Work partially supported by the Spanish Ministry of Science and Technology under the Research Project PRIVILEGE (TIC2003-08184-C02-01)

It is well known that one of the most relevant advantages of WS is that they can easily be accessed over the Internet, using ubiquitous web protocols and data formats such as XML. Nevertheless, simplicity of access makes them vulnerable to various security threats. Valuable corporate information, applications and systems are exposed when WS are used to provide access to critical business functions. Currently, access control mechanisms for WS, if present, are the same as for web pages, ignoring the security requirements. A secure solution for WS is not a trivial issue. The reasons are that WS are intrinsically dynamic, heterogeneous in implementation and security requirements, and decentralized in architecture and administration.

The Semantic Web, also known as the *Internet of meanings*, is not just a vision of the future (Berners-Lee, 2000). The next step is to implement this new vision of the Web as real-world applications (W3C, 2002). In this way, this paper introduces the description and implementation of a model for WS access control, which is built on the basis of (i) the separation of the authorization and access control management responsibilities and (ii) the extensive use of semantic information, in order to achieve security and interoperability. The proposed solution is scalable, facilitates the management of the access control system and enables the semantic integration and interoperability of heterogeneous WS. We introduce the *Semantic Policy Language* (SPL) for the specification of access control criteria based on the use of *attribute certificates*. We also present a solution to integrate *Privilege Management Infrastructures* (PMIs) and to facilitate administration tasks based on semantic information. More precisely, semantic information is used in our approach not only for the integration of the external PMI, but also for the semantic validation of policies and the dynamic instantiation of parameters of the policies based on properties of the resources accessed.

The structure of this paper is as follows. Section 2 provides a background on the security definitions of the basic terminology used in this work. Additionally, it provides background of WS security issues, and reviews access control models based on attribute certificates. Section 3 analyses related work. Section 4 describes in detail our proposal. Finally, section 5 concludes the paper.

2 Background

2.1 Security terminology

For a better understanding of the paper we briefly define in this subsection some security-related terms that are frequently used throughout the sections of this work. These definitions has been partially adapted and refined for this paper from those definitions included in the multiple Recommendations of the *International Telecommunication Union – Telecom Standardization*

(ITU-T) [*]. The list of terms, which try to follows a logical order, as well as their definitions, follows:

- *Security Service*: A service, provided by a layer of communicating open systems, which ensures adequate security of the systems or of data transfers.

- *Authentication*: Provision of assurance of the claimed identity of a user. This includes the process of verifying the claimed identity of one user to another user.

- *Authorization*: Granting of rights, what includes granting of access based on access rights or privileges. It implies the rights to perform some operation, and that those rights or privileges have been granted to some process, entity, or human agent.

- *Access Control*: Prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner. This means that there will be a limitation of the flow of information from the resources of a system only to authorized users.

- *Authority*: An entity responsible for the issuance of certificates.

- *Credential*: Data object that is transferred to establish the claimed identity of a user.

- *Public Key Certificate* (or *identity certificate*): A data structure, digitally signed by a Certification Authority, which binds the identity of a user with his/her public key.

- *Certification Authority (CA)*: A trusted organization that accepts certificate applications from users, authenticates applications, issues certificates and maintains status information about certificates.

- *Public Key Infrastructure (PKI)*: A process for issuing public key certificates, which includes standards, Certification Authorities, communication between authorities and protocols for managing certification processes.

- *Attribute Certificate*: A data structure, digitally signed by an Attribute Authority, which binds some attribute values with identification information about its holder.

- *Attribute Authority (AA)*: An authority trusted by one or more users to create, sign and issue attribute certificates which assigns privileges.

- *Source of Authority (SOA)*: An Attribute Authority that a privilege verifier for a particular resource trusts as the ultimate authority to assign a set of privileges.

[*] http://www.itu.int/ITU-T/

- *Privilege Management Infrastructure (PMI)*: The infrastructure able to support the management of privileges in support of a comprehensive authorization service and in relationship with a Public Key Infrastructure.

2.2 Web Services Security

As mentioned in the introductory section, WS are supported by the combination of three standards: SOAP, UDDI and WSDL. SOAP is used to interact with WS, UDDI is used to publish them, and WSDL is used to describe their functionality. However, other issues, like security requirements, have not been considered in these standards. WS demand a security framework, especially for access control, that does not inhibit the exchange of data, what is essential for their success. Languages such as WSDL represent a valuable tool for the description of functionality but not for security properties. This is reasonable, since WSDL is intended for clients to learn what WS provide. On the other hand, a mechanism to describe and enforce security requirements is not intended for clients, but for the access control system. Furthermore, functional descriptions are public while security properties and access control policies are usually confidential.

Nowadays, security is the main obstacle for the adoption of WS in corporations. Existing security solutions for WS are supported by technology that has been in use for years in other scenarios. In fact, the most common approaches are to use a standard firewall system with *Secure Sockets Layer* (SSL) or *Transport Layer Security* (TLS) (Dierks, Allen, 1999), and to build a proprietary gateway for authentication and authorization. Undergoing work in the XML area is aimed to provide more control than SSL. As a matter of fact, different XML standardization bodies such as IETF, OASIS, and W3C are working on the specification of security protocols usable with WS, including extensions to encryption, digital signatures and non-repudiation. Table 1 illustrates some of the proposed standards.

Standard	***Body***	***Status***	***Brief Description***
eXtensible Access Control Markup Language (XACML)	OASIS	Draft	Expresses policies for information access
Security Assertion Markup Language (SAML)	OASIS	Mature	Exchanges authentication and authorization information
eXtensible Rights Management Language (XrML)	OASIS	Draft	Manages copyrights of digital content
XML Digital Signature	W3C	Mature	Provides integrity, signature assurance and non-repudiation
XML Encryption	W3C	Mature	Encrypts and decrypts digital content
X.509 2000	ITU-T	Mature	Provides frameworks for PKIs and PMIs

Table 1. Some Standards for Security in WS.

Recently, IBM, Microsoft, VeriSign and OASIS have started cooperation in the specification of *Web Services Security* (OASIS, 2002). This initiative describes SOAP enhancements to provide protection of messages. WS-Security also provides a general-purpose mechanism for the association of security tokens with messages and describes how to encode binary security tokens such as X.509 certificates (ITU, 2000). Although WS-Security does not address security issues such as authorization or access control, it represents an important initiative to support other security services.

In order to meet the security requirements that WS demand, additional mechanisms are needed. As previously stated, authorization and access control are critical issues because of the specific characteristics of WS. In this environment, and besides controlling access to information, the control to the operations that the Web service offers must be applied. Procedures based on centralized control do not represent a suitable solution because of the distributed nature of WS. A single control point represents a weak feature for security attacks and fault tolerance, among other disadvantages. It also reduces system performance because it introduces a bottleneck for request management. Additionally, most of times it enforces homogeneous access control schemes that do not fit well in heterogeneous user groups and organizations.

WS facilitate the integration with third parties including suppliers, customers and partners, making essential that access rights are tightly controlled and kept up-to-date. However, and because multiple parties are involved, it is often difficult or impossible to agree on a common access control scheme. Hence, access control administration is an increasing challenge for WS environments.

2.3 Access Control based on Attribute Certificates

Based on asymmetric cryptography (Diffie, Hellman, 1976), digital certificates are used to bind a public key to some information. *Identity certificates* (a.k.a. *public-key certificates*) (ITU, 1997) are the most common type of digital certificates in use. They bind identity information to keys. On the other hand, *attribute certificates* (ITU, 2000) bind attributes to keys. Among other applications, they provide means for the deployment of scalable and flexible access control schemes, since access conditions are expressed in terms of sets of attributes instead of users or groups. Users must provide attribute certificates attesting that they meet the requirements. Opposed to traditional access control schemes, a high number of users and attributes do not degrade performance and manageability of this solution. Suppose John Doe is an authorized broker at the Chicago Board of Trade. Then John will have two separate certificates: an identity certificate attesting his identity information and an attribute certificate attesting he is an authorized broker at the Chicago Board of Trade. Both certificates can be related, for instance, by including the serial number and/or hash value of the identity certificate in the attribute certificate.

One of the main advantages of attribute certificates is that they can be used for various purposes. They may contain group membership, role, clearance, or any other form of authorization. A very essential feature is that attribute certificates can securely transport authorization information in distributed applications. This is especially relevant because, through attribute certificates, authorization information becomes “mobile”, which is highly convenient for scenarios such as WS. This mobility provides the foundation for a better alternative to actual *Single Sign-On* schemes (Sundsted, 2002).

The mobility feature has been used in applications since the publication of the 1997 ITU-T X.509 Recommendation. However, it has been used in a very inefficient way. That Recommendation introduced an ill-defined concept of attribute certificate that was not independent of the identity certificate. To be more precise, when using that solution, the change of privileges indirectly forces a costly revocation of the identity related information. Moreover, that solution does not solve delegation and impersonation issues, which are especially relevant in many applications. The ITU-T 2000 Recommendation provides a more suitable solution because it clearly defines a framework, a *Privilege Management Infrastructure*, or PMI, where identity and attribute certificates, although related, can be independently managed.

If we focus simultaneously on security, scalability, interoperability and mobility, it is advantageous to separate the responsibilities of access control management from certification of attributes. There are some reasons for this statement. In centralized access control schemes each application requires its own database or directory of authorizations, which must be administered and maintained. The result is that for every user, identities and profiles must be entered multiple times and synchronized dynamically, increasing the operating costs associated with change management and making the process cumbersome. However, the same users’ attributes are often used in multiple access decisions in different systems. We can conclude that users’ attributes can be shared by all access control systems, while access criteria are specific.

Suppose now that our friend John Doe is also member of the Chicago Siesta Club (CSC), a public library, Greenpeace, etc. If centralized access control schemes are used in these institutions, each one will have to locally register the different attributes of John Doe that are applicable to their access control policies. For instance, if the CSC has a discount for Greenpeace members then it is necessary that the membership of John to Greenpeace is recorded in the local users database of CSC. How can CSC be sure that John is member of Greenpeace? What if John leaves Greenpeace? How does CSC know about this? On the contrary, if the attribute certification function is separated then access control systems responsibilities are limited to establishing the local access control policies, making the system simpler, more dynamic and flexible, and more secure. Obviously, this approach requires that the access control system is complemented by an external component providing certification functions. Precisely, the PMI is that component. In this framework, several entities called *Source of Authorization* (SOA), are responsible for issuing attribute certificates. Typically, every SOA issues certificates regarding a small number of semantically related attributes.

A consequence of the separation of access control and authorization functions (now provided by the PMI) is that the access control administrators do not have control over some factors that are used in their access control systems. Consequently, a mechanism to establish the trust between these administrators and the PMI is required. As we will show later, we have addressed this problem using semantic information about the certifications issued by each SOA. This assists the security administrators in the creation and semantic validation of access control policies.

In the case of access control systems for WS, the integration of an external PMI represents a step towards the solution of problems such as scalability, interoperability and separation of duties. The semantic integration is the best approach to interoperability, as it allows the description and exploitation of valuable information. This is a very interesting application scenario for semantic modelling. Its relevance is derived from the security requirements of the environment and the necessity for access control systems to understand the semantics of the attribute certificates managed by the PMI.

3 Related Work

Recent literature in the area of access control for distributed heterogeneous resources from multiple sources shows the use of attribute certificates and PMIs. Firstly, we highlight two research projects, Akenti (Thompson, 1999) and Permis (Chadwick, 2002). Akenti Project proposes an access control system to restrict access to distributed resources controlled by multiple stakeholders. The requirement for the stakeholders to trust the rest of the servers in the network, as well as some security vulnerabilities related to the existence of positive and negative use-conditions, are the main drawbacks of Akenti. On the other hand, the objective of Permis is to set up an integrated infrastructure to solve identification and authorization problems. After a careful study of various policy languages, Permis researchers have concluded that XML is the most appropriate candidate to be used as a language for specification of policies. Permis has the same limitations than RBAC because it is based on this access control model.

Role based access control is commonly accepted as the most appropriate paradigm for the implementation of access control in complex scenarios. RBAC can be considered a mature and flexible technology. Numerous authors have discussed the access properties and have presented different languages and systems that apply this paradigm (Sandhu, Coyne, Feinstein, Youman, 1996) (Sandhu, Ferraiolo, Kuhn, 2000). The hierarchical RBAC model is a more sophisticated version of the simple RBAC model. With this model, the roles are organised hierarchically and the specialized roles inherit the privileges of the more general roles. A common hierarchy in this model can be the following one:

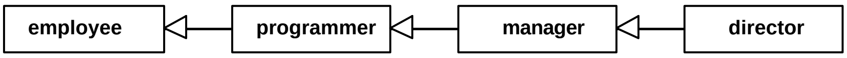

Fig. 1. A simple role hierarchy

If certain privilege is assigned to an employee role, possession of any of the superior (more specific) roles enables the same privilege, even though the role specification does not state this explicitly. For example, if a programmer is given permission to enter the computer building, managers and directors would also inherit this permission.

However, very dynamic systems with high volume of heterogeneous data, like semi-structured data systems, require more flexible constructions for the expression of access control policies. In RBAC, the security administrator defines the structure of groups, which is usually assumed to be static. Although the grouping of users can suffice in many situations, it is not flexible enough to cope with the requirements of more dynamic systems where the structure of groups can not be anticipated by the administrators of the access control system. In these scenarios new resources are incorporated to the system continuously and each resource may possibly need a different group structure and access control policy. Furthermore, the policy for a given resource may change frequently, sometimes requiring a change in the structure of roles.

Likewise, other traditional access control schemes such as *Mandatory Access Control* (MAC), *Discretionary Access Control* (DAC) are not appropriate for scenarios where the users of a system are previously unknown or with a very large number of registered users. In these systems, and for scalability reasons, it is not practical to keep access information for each user.

Consequently, a different approach is required in order to solve the scalability problems of these systems, to facilitate access control management and to provide means to express access conditions in a natural and flexible way. Our diagnostic is that the main problem with role based access control is that the model is built on three predefined concepts: “user”, “role” and “group”. The definition of roles and the grouping of users can facilitate management, especially in corporation information systems, because roles and groups are easily identifiable and fit naturally in the context of the organizational structures of the companies. However, when applied to some new and more general access control scenarios, these concepts are somewhat artificial. In fact, it is frequent that a classification of objects is first made and then roles are defined in a direct mapping to the different classes.

We believe that a more general approach is needed for these new environments. For example, in the referred situations, groups are an artificial substitute for a more general tool: the attribute. In RBAC, groups are usually defined based on the values of some specific attributes (employer, position, …). Some attributes are even built into most of the current access control models. This is the case of the "user" element; the identity is just one of the most useful attributes, but it is not necessary in all scenarios and, therefore, it should not be a built-in component of a general model. Furthermore, there are scenarios where the use of identity must be avoided. Finally, access control models must take into account that the creation and maintenance of access control policies is a difficult and error prone activity. Therefore, these models must be designed to facilitate and guarantee the correct administration of the system.

An orthogonal problem is the association of actions that must be executed before access to a resource is granted. This concept, introduced in (Kudo, Hada, 2000), is known as *Provisional Authorization* or *Provisional-Based Access Control*, PBAC. In a PBAC system the client must complete a set of actions (such as notification to the resource owner, online authorization or payment) in order to gain access to a resource.

The *Semantic Access Control* (SAC) model (Yagüe, 2003) provides an appropriate solution to aforementioned problems, especially for heterogeneous, distributed and large environments. Precisely, this paper shows suitability of SAC model to the WS scenario. The SAC model is based on the semantic properties of the resources to be controlled, properties of the clients that request access to them, semantics about the context and finally, semantics about the attribute certificates trusted by the access control system. The SAC model has been implemented on the basis of the Semantic Policy Language (SPL) to specify the access control criteria, and the semantic integration of an external authorization entity.

Different XML-based languages have been proposed for access control, digital rights management, authentication and authorization. Although many similarities and interesting features can be found among them, some other features, such as policy parameterisation and composition are not supported. Moreover, some features provided by those languages are not appropriate in WS scenarios. The most relevant is XACML (OASIS, 2003), an OASIS standard that proposes two XML-based languages to describe access control policies and access decision requests and responses. Although XACML and SAC share some similarities, there are important differences, such as:

1. In the XACML Specification the term attribute is used in place of the terms group and role. In SAC, the term attribute is generic, and can be used to represent any kind of

property of the access requester (application, web service, user…) or the resource to be accessed.

2. Separate XACML policies can be combined into a single policy. XACML provides means to specify precise procedures for combining the results of the evaluation of the basic policies. The rule-combining algorithm defines a procedure for arriving at an authorization decision given the individual results of evaluation of a set of rules. Some predefined algorithms are included for: deny-overrides, permit-overrides, first applicable and only-one applicable. Summarizing, policy composition in XACML is limited to the combination of partial access decisions. On the other hand, SAC policies are built on the basis of semantics of the access control criteria. Consequently, the composition of different access control policies is performed on the basis of the semantics of these policies, allowing rich combination of policies, not only of partial access decisions.

3. Allocation of policies to resources in XACML is explicit and static. Opposite to this, SAC defines a mechanism for the dynamic allocation of policies to resources, based on semantics of the latter.

4. XACML provides facilities for content-based access when the information resource can be represented as an XML document. SAC supports content-based naturally imposing no restriction on the format: every kind of resource (a physical resource, a digital document with any format …) can have an associated document, describing its semantics.

5. The architecture of XACML is one of its main contributions. XACML proposes a very flexible scheme based on the definition of *Policy Enforcement Points* (PEPs), *Policy Decision Points* (PDPs), etc. The fully distributed and open approach of SAC makes possible that the inclusion of PEPs and PDPs does not require any modification on the SAC model. As with the case of XACML, SAC policies may be written and analyzed independently of the specific environment in which they are to be enforced.

6. Finally, the SAC model considers the execution of some actions during the enforcement of the access request, therefore providing full support for Provisional-Based Access Control (PBAC). XACML only supports a predefined set of these actions.

Two other relevant access control systems using XML are Author-X (Bertino, Castano, Ferrari, 2002) and FASTER (Damiani, De Capitani, Paraboschi, Samarati, 2002a). Because both systems have been specifically developed to control access to XML documents, they do not fit naturally in the WS environment. While Author-X policy language uses DTDs, our Semantic Policy Language and FASTER use XML-Schema. Both Author-X and FASTER define hierarchic access control schemes based on the structure of the document, opposite to SAC which is based on semantics of contents. While SAC provides secure content distribution and originator control by means of active containers (Lopez, Maña, Pimentel, Troya, Yagüe, 2002), the FASTER system does not support any content protection mechanism. On the other hand, content protection in Author-X is founded on the concept of (passive) secure container, which introduces disadvantages from the point of view of security and access control system management. Author-X is based on credentials that are issued by the access control administrator. Therefore, in practice, each credential will be useful only for a single source, limiting interoperability. A direct consequence of this approach is that users must subscribe to sources before they can access their contents. The FASTER approach differs from ours in that it is completely 'server-side'. Authorizations can be specified at document or instance level (in XML documents), or alternatively at schema level (in Document Type Definition -DTDs-). Authorizations specified in a DTD are applicable to all XML documents that conform to it.

In the WS environment, operations are the main target of authorization, and this is the reason why both systems do not fit naturally into this environment. The access control scheme described in FASTER has been applied to the XML structure of SOAP calls (Damiani, De Capitani, Paraboschi, Samarati, 2002b). This access control is based on user groups, roles and physical locations, following the technique of defining a subject hierarchy. However, this approach is not adequate for scenarios where the structure of groups can not be anticipated. Moreover, in the WS area, new services are incorporated dynamically and each one may need a different group structure and access control policy. Furthermore, the policy for a given Web Service may change frequently.

Another characteristic of the work described in FASTER is the use of hierarchies and the propagation of the authorizations of a group to all its members, of a role to all its specializations and of a location pattern to all the machines in its subnetwork. As a consequence, for the general case of WS with non-basic security requirements, the number of different authorizations (positive and negative) to be defined grows exponentially. In fact, negative authorizations granted to roles are unreliable as the authors admit in (Bertino, Castano, Ferrari, 2002).

4 Description of the Semantic Access Control System

Based on the idea of separating the access control function from the authorization procedure (credential issuance or attribute certification), we propose the integration of an external PMI supported by semantic information about the certification entities. Through our proposed solution, semantic and contextual validation of policies is made possible.

This section is divided into three subsections, as follows. First subsection presents the architecture of the access control system. The second subsection is devoted to the description of the different components of SPL. The last subsection discusses policy management and, one of the most relevant contributions of our proposal, policy validation.

4.1 Architecture of the Access Control System

A general overview of the main components of the system and their relationship is depicted in Figure 2. It includes three different components: the WS client, the WS server and the PMI.

Several PMI nodes, some of which are SOAs, conform the PMI. Every SOA produces and digitally signs a set of *Source Of Authorization Descriptions* (SOADs) that express the semantics of the attribute certificates it issues. These metadata documents describe the different attributes certified by a SOA, including names, descriptions and relations of attributes. SOADs are used to establish the trust between the PMI and the access control systems. They convey the information needed by the access control system to understand the semantics of the attribute certificates, which is essential in order to take appropriate access decisions. The information contained in SOADs is also essential for the semantic validation of the policies, enabling the detection of semantically incomplete (or incorrect) policies. In fact, the set of SOADs represents the semantic description of the PMI. Full integration of the PMI can be achieved transparently for the rest of the system based on this description.

The WS Server includes several components related to the access control. An access control proxy, *AC Proxy*, is included in order to provide a transparent access control service for both WS clients and developers. The AC Proxy intercepts the calls to the WS. However, application-level access control is sometimes an important requirement. For security-aware WS the proxy might not be required. The last component, called *Policy Assistant* is responsible for the management and validation of the access control criteria, defined by SPL specifications. These specifications are modularly described using SPL *policies*, *Policy Applicability Specifications* (PAS), *Secured Resource Representations* (SRR), and the *context* metadata.

After receiving an access request, the AC Proxy forwards the authorization request to the *Access Control* module and retrieves the attribute certificates from the PMI through the *PMI Client*. The PMI Client retrieves the SOADs using the *SOAD Server* to determine which PMI node must be contacted. The Access Control module is responsible for producing access decisions by performing dynamic allocation and policy evaluation. When a request is received, it analyses the semantic metadata available for the target resource contained in the SRR, finds the appropriate PAS and retrieves the necessary SOADs. Using this information, the Access

Control is able to find the applicable policies. These policies are then analysed and instantiated using the metadata about the resource, SRR, and the context. Finally, all policies are combined and evaluated producing an access decision that is returned to the AC Proxy. This process is called *dynamic policy allocation*.

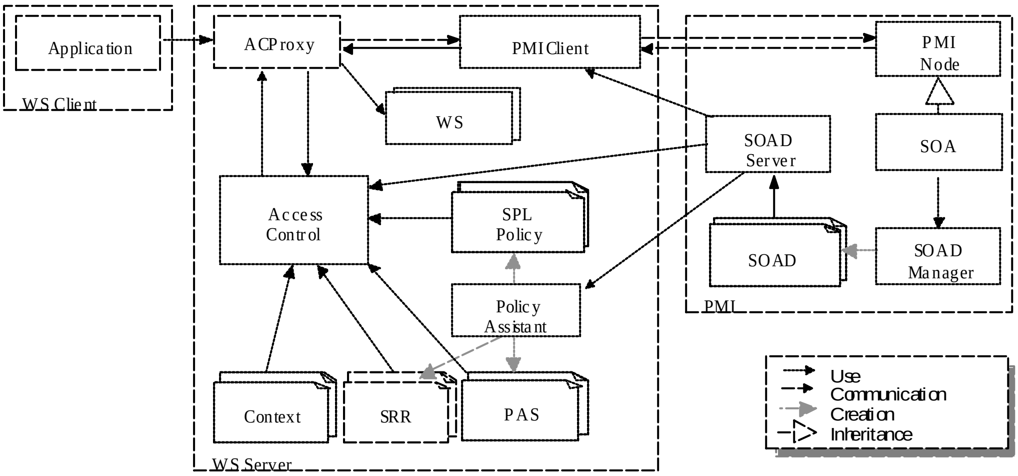

Fig. 2. Overview of the system

4.2 The Semantic Policy Language

As we mentioned in section 2, other XML-based languages have been developed for access control and authorization. However, their generality results in a high complexity. Furthermore, many of their features are not useful in WS scenarios. On the other hand, some important features of SPL are not considered in these languages. For this reason, we have developed a specific XML-based language to specify the access control policies. This language is called Semantic Policy Language because it is based on semantic properties about the resources to be accessed, the PMI and the context. These semantics are used during the specification of access control criteria, dynamic policy allocation, parameter instantiation and policy validation.

The definition of access control policies is a complex activity that presents many similarities with computer programming. Thus, SPL includes some of the mechanisms used there in order to reduce the complexity, such as modularity, parameterisation and abstraction. Additionally, and as stated, our solution is based on the modular definition of policies in order to provide the simplicity and flexibility required by complex systems. Modularity in our solution implies:

i) Separation of specification in three parts: access control criteria, allocation of policies to resources and semantic information (properties about resources and context).

ii) Abstraction of access control components.

iii) Ability to reuse these access control components.

iv) Reduction of the complexity of management due to previous properties.

Moreover, the use of semantic information about the context allows the administrator to include contextual considerations in a transparent manner, while helping the semantic validation task too.

Usual components of access policies include the target resource, the conditions under which access is granted/denied and, sometimes, access restrictions. As opposed to other languages, specifications in SPL do not include references to the target object. Instead, a separate specification called *Policy Applicability Specification* (PAS) is used to relate policies to objects dynamically when a request is received. Both SPL Policies and PAS use semantic information about resources, included in SRRs, and other contextual information documents.

SPL Policies and PAS can be parameterised allowing the definition of flexible and general policies, thus reducing the number of different policies to manage. Parameters, which can refer to complex XML elements, are instantiated dynamically from semantic and contextual information.

Additionally, policies can be composed importing components of other policies without ambiguity. This compositional approach allows us to define the abstract meaning of the elements of the policies, providing a mechanism to achieve abstraction, which also helps in reducing the complexity of management. Tools developed to graphically manage the relations among policies, as well as with other components, are also essential for a simple and flexible management. The schema for SPL specifications is represented as a set of XML-Schema templates that facilitate the creation of these specifications, allowing their automatic syntactic validation.

Figure 3 shows the conceptual model of the SPL language. SPL policies can include components defined locally as well as imported elements. The ability to import elements enables the modular composition of policies based on the XPath standard. An SPL Policy is composed of a set of *access_Rule* elements. Every *access_Rule* defines a particular combination of attribute certificates required to gain access, associated with an optional set of actions (such as *Notify_To*, *Payment* and *Online_Permission*) to be performed before access is granted. In this way provisional authorization or PBAC is enabled in SPL.

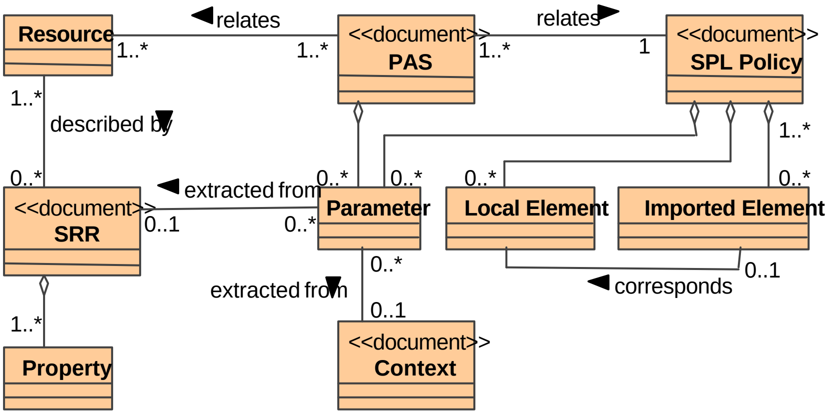

Fig. 3. Conceptual model of the SPL Language

Figure 4a shows an example of an SPL policy requiring an attribute certificate stating the client is an authorized broker. This policy has only one access rule indicating that access should be granted to all brokers authorized by the Chicago Board of Trade administration authority. Any attribute certificate that is proved equivalent to this one will be accepted because the `Equivalence` attribute of the `spl:attribute` tag is set to `“Enabled”`. Also notice that no information regarding the reason why the request is denied will be given to users that do not meet the access criteria because this access rule is not public (the `Public` attribute of the `spl:access_Rule` tag is set to `“false`”. This feature is used by access control administrators to avoid unauthorized users to learn about the existing access policies.

The SRR is a simple and powerful mechanism to describe properties about resources. Properties described in SRRs are used for the instantiation of policies and PAS, and to locate the applicable policies. An example of a SRR is included in Figure 4b. Dynamic allocation of policies to resources is a very flexible and useful mechanism that solves the problem of associating policies to newly created objects. The use of dynamic policy allocation needs a rich set of metadata about the resources. This semantic meta-model is used to locate the right policy for each resource, based on its relevant properties.

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:policy xmlns:spl="http://www.lcc.uma.es/WS"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.lcc.uma.es/WS PolicyTemplate.xsd">
    <spl:access_Rules>
        <spl:access_Rule Name="Auth_Brokers" Public="false ">
            <spl:attribute_Set>
                <spl:attribute attributeID="Auth_Broker" Equivalence="Enabled">
            <spl:attribute_Name>Function</spl:attribute_Name>
            <spl:attribute_Value>Broker</spl:attribute_Value>
            <spl:SOA_ID>CBOT_ADMIN</spl:SOA_ID>
            <!-- Chicago Board Of Trade Administration -->
                </spl:attribute>
            </spl:attribute_Set>
        </spl:access_Rule>
    </spl:access_Rules>
</spl:policy>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:SRR xmlns:spl="http://www.lcc.uma.es/WS"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.lcc.uma.es/WS SRR_WS.xsd"
      resource="http://www.uma.es/ConsultingWS[Profit_Classify]">
<!--  AC properties -not public- about the Profit_Classify Operation of the Consultancy Web
          Service –>
   <spl:property>
   <spl:property_Name>responsible</spl:property_Name>
   <spl:property_Value>admin@consulting.com</spl:property_Value>
   </spl:property>
   <!-- e-mail of the administrator –>
   <spl:property>
   <spl:property_Name>scope</spl:property_Name>
   <spl:property_Value>local</spl:property_Value>
   </spl:property>
   <!-- does not access external resources –>
</spl:SRR>
```

Fig. 4. (a) Consulting_Access.xml Policy and (b) SRR Example

The PAS provides an expressive way to relate policies to resources, either explicitly or based on the metadata about the objects (e.g. type of content, owner, price, etc.). PAS documents include three main elements: policy, objects and instantiation. The *policy* element indicates which policy is applicable to the specified *objects*. These are defined specifying their *location* and the *conditions* to be fulfilled by the semantics of these objects (SRRs). Optionally, *operation* elements can be used to define which operations of the target object are controlled by the declared policy, allowing a finer grained access control. In case no operation element is included, the policy is applicable to all of the object operations. The instantiation element describes the mechanism to instantiate parameters in the policies. Figure 5 shows an example of applicability rules indicating that the *Consulting_Access.xml* policy is applicable to all WS of type 'Investment' at *http://www.lcc.uma.es/ConsultingWS*.

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:PAS xmlns:spl="http://www.lcc.uma.es/WS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="http://www.lcc.uma.es/WS pas.xsd">
<!-- 'Consulting_Access' is applicable to the Profit_Classify operation of all the Web Services of type 'Investment' in
            'http://www.lcc.uma.es/ConsultingWS/' -->
  <spl:policy>http://www.uma.es/WS/Consulting_Access.xml</spl:policy>
  <spl:object>
     <spl:object_Location>http://www.lcc.uma.es/ConsultingWS</spl:object_Location>
     <spl:operations>
          <spl:operation>Profit_Classify</spl:operation>
     </spl:operations>
     <spl:conditions>
        <spl:condition predicate="equals">
          <spl:property_Name>WS_Type</spl:property_Nam e>
          <spl:property_Value>Investment</spl:property_Value>
        </spl:condition>
     </spl:conditions>
  </spl:object>
</spl:PAS>
```

Fig. 5. PAS for the Consulting_Access.xml

4.3 Policy Management and Validation

The creation and maintenance of access control policies is a difficult and error prone activity. The Policy Assistant component is designed to help administrators to specify those policies and validate them in order to find syntactic and semantic errors. It includes components for the automated validation of policies at different levels. The syntax of SPL policies is validated against the corresponding XML Schema. We have developed a specific *Semantic Policy Validator* (SPV) as part of the Policy Assistant component to perform different types of semantic validations. These validations are supported by the semantic information defined by means of XML metadata.

An interesting feature of the SPV is that allows policies to be validated in the context where they will be applied. Policy context validation is based on the SOADs and the *Context* metadata. The higher expressiveness of SPL specifications, along with the additional semantic information in the form of XML metadata, allows an easy semantic integration of our access control system with a PMI. Additionally, it enables interoperability among access control mechanisms of different WS.

To illustrate the context and semantic validation of policies, let us consider an investment adviser application that accesses different WS offered by consulting firms. Those WS classify a series of products according to their estimated profitability. Clients use the application to take investment decisions regarding different products, based on the classifications received from the different consulting firms.

Each web service must grant access to authorized brokers, while prohibiting the access to competing firms. This is a case where the use of negative authorizations seems reasonable. Where positive authorizations specify permissions for an access, negative authorizations specify denials for an access. In access control systems based on credentials or attribute certificates, negative authorizations represent a problem because a user can avoid being subjected to a negative authorization simply by not presenting the corresponding certificate. Solutions so far are unable to solve this problem because no information about the context is considered (Yagüe, 2002).

Our approach uses semantic information about the context to solve this situation without using negative authorizations. For example, the context metadata can state that each authorized broker is either a member of a partner firm or a member of a competing firm. This type of implicit information is not always obvious for the administrator. Therefore, based on context metadata, the access control administrator can realize that the policy must require membership in a partner company for granting access.

Figure 6a shows the resulting policy. This policy checks that the user is an authorized broker importing the corresponding attribute from the policy defined in Figure 4a. It declares a parameter called *Company*. Notice that, when instantiated, parameter references will be resolved to attributes of the actual parameter. The PAS for this policy is shown in Figure 6b. Instantiation criteria for the Company parameter are declared in this document. The actual parameter is an XPath reference to a list of partners stored as semantic information about the context.

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:policy xmlns:spl="http://www.lcc.uma.es/WS"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.lcc.uma.es/WS
     http://www.uma.es/WS/PolicyTemplate_WS.xsd">
   <spl:parameter>Company</spl:parameter>
   <spl:access_Rules>
      <spl:access_Rule Name="Partner_Member" Public="false">
         <spl:attribute_Set>
            <spl:attribute predicate="equals">
                <spl:attribute_Name>
                    Member
                </spl:attribute_Name>
                <spl:attribute_Value>
                    *Company[@name]
                </spl:attribute_Value>
                <spl:SOA_ID>
                    *Company[@SOA_ID]
                </spl:SOA_ID>
            </spl:attribute>
         <spl:import Url="Consulting_Access.xml"
    XPath="//attribute[@attributeID='Auth_Broker']"/>
         </spl:attribute_Set>
      </spl:access_Rule>
   </spl:access_Rules>
</spl:policy>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:PAS xmlns:spl="http://www.lcc.uma.es/WS"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.lcc.uma.es/WS_pas.xsd">
<!-- 'MemberShip_Partner' is applicable to the 'Profit_Classify' operation of Web Services of type
   'Investment' in 'http://www.lcc.uma.es/ConsultingWS/' -->
   <spl:policy>
      http://www.uma.es/WS/Partner_Membership.xml
   </spl:policy>
   <spl:object>
      <spl:object_Location>
         http://www.lcc.uma.es/ConsultingWS
      </spl:object_Location>
      <spl:operations>
            <spl:operation>Profit_Classify</spl:operation>
      </spl:operations>
      <spl:conditions>
         <spl:condition predicate="equals">
            <spl:property_Name>WS_Type</spl:property_Name>
            <spl:property_Value>Investment</spl:property_Value>
         </spl:condition>
      </spl:conditions>
   </spl:object>
   <spl:instantation>
      <spl:formal_Parameter>Company</spl:formal_Parameter>
      <spl:actual_Parameter path="Partners_List//Company">
         http://www.uma.es/WS/SOAD/MemberShip_Partner_Context.xml
      </spl:actual_Parameter>
      <!--Partners_List contains names and SOADs of our partners –>
   </spl:instantation>
</spl:PAS>
```

Fig. 6. (a) Partner_Membership.xml Policy and (b) its PAS

The semantics of the policies depend heavily on the semantics of the attribute certificates in those access control schemes that are based on that type of certificates. In the SPL access control model the semantics of the attribute certificates are stated in SOADs.

The ability to perform a semantic validation of access control policies is an essential design goal of this access control system. Both the SPL language and the semantic description of the certificates issued by each SOA (conveyed by SOAD documents) are designed to serve this objective. The semantic validation ensures that the policies written by the administrator produce the desired effects. The SPV can perform three types of validations:

- *Test Case Validation:* Given a request to access a resource and a set of attribute certificates, this algorithm outputs the sets of attribute certificates needed for accessing that resource. Most of times this feature will be used to check that a set of attribute certificates is incompatible with the access criteria for that resource. For instance, the administrator of our university can use this validation to guarantee that it is not possible for a student to access a given resource. During the validation process, the SPV generates the sets of attribute certificates that are not excluded by the input set, and checks the generated ones against all possible combinations of attribute certificates that grant access to the resource.

- *Access Validation:* Given a request to access a resource, this algorithm outputs the sets of certificates that grant access to that resource. For this validation process, the SPV generates the policy for the resource and all sets of attribute certificates equivalent to those required by the policy.

- *Full Validation:* The goal of this process is to check which resources can be accessed given a set of attribute certificates. Therefore, SPV generates the policy for each resource and, afterwards, all attribute certificates that can be derived from the input set of attribute certificates. Finally, it informs of every resource that can be accessed using the input attribute certificate set.

5 Conclusions and Future Work

We have presented a system that provides interoperable and distributed access control for WS. We have addressed the integration of a separate Privilege Management Infrastructure by defining mechanisms for the semantic description of its components. These mechanisms allow us to seamlessly integrate the authorization entities in our system. The Semantic Policy Language, an XML-based policy definition language designed to specify policies in a simple way and to facilitate semantic policy validation, has been introduced. SPL specifications are modular and can be composed without ambiguity. Finally, we also have addressed the problem of the association of policies to resources (WS or their operations) in a dynamic, flexible and automated way.

An important feature is the extensive use of XML metadata, which facilitates the security administration and enable interesting functionalities of the system such as the contextual semantic validation of policies. Metadata are applied at different levels in our proposal. On the one hand, access control policies benefit from metadata for its creation and semantic and contextual validation. Likewise, resources have metadata associated that are used for the dynamic policy assignment and parameter instantiation. Additionally, metadata are used for the specification and acquisition of certification rules. On the other hand, metadata is an essential tool for the integration of the external PMI in the access control system.

Current implementation includes functional versions of all policy-management and enforcement applications. All elements necessary for the semantic integration of the external PMI (SOAD Manager, SOAD Server, PMI Client) have also been developed. A beta version of the *ACProxy* component is available. Development process is currently centered on the finalization of this component. Ongoing work is focused on achieving a richer "use control" for some types of WS (for instance, we are working on the integration of a pay-per-use model).

References

Berners-Lee, T. (2000), "Semantic Web". Available http://www.w3.org/2000/Talks/1206-xml2k-tbl/Overview.html.

Bertino, E., Castano, S., Ferrari, E. (2002), "Securing XML documents with Author-X", IEEE Internet Computing, vol. 5 n. 3, pp.21-31.

Chadwick, D. W. (2002), "An X.509 Role-based Privilege Management Infrastructure. Business Briefing", Global Infosecurity, Available http://www.permis.org/

Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P. (2002a), "A Fine-Grained Access Control System for XML Documents", ACM Transactions on Information and System Security, vol. 5 n. 2, pp. 169-202.

Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P. (2002b), "Securing SOAP E-Services", International Journal of Information Security, vol. 1 no. 2, pp. 100-115.

Dierks, T., Allen C. (1999), "The TLS Protocol Version 1.0. IETF RFC 2246". Available http://www.ietf.org/rfc/rfc2246.txt

Diffie, W., Hellman, M. (1976), "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. 22 n. 6., pp. 644-654.

ITU (1997), "ITU-T Recommendation X.509, Information Technology – Open systems interconnection – The Directory: Authentication Framework". Available http://www.itu.int/rec/recommendation.asp?type=folders&lang=e&parent=T-REC-X.509

ITU (2000), "ITU-T Recommendation X.509: Information Technology - Open systems interconnection – The Directory: Public-key and attribute certificate frameworks", Available http://www.itu.int/rec/recommendation.asp?type=folders&lang=e&parent=T-REC-X.509

Kudo, M., Hada, S. (2000), “XML Document Security based on Provisional Authorisation”, 7th ACM Conference on Computer and Communications Security.

Lopez, J., Maña, A., Pimentel, E., Troya, J.M. and Yagüe, M.I. (2002), “An Infrastructure for Secure Content Distribution”, 4th. International Conference on Information and Communications Security, LNCS 2513, Springer-Verlag.

OASIS (2000), Universal Description, Discovery and Integration. Available http://www.uddi.org/specification.html

OASIS (2002), “Web Services Security”. Available http://www.oasis-open.org/committees/wss/

OASIS (2003), “XACML 1.1 Specification Set”, Available http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

Sandhu, R.S., E.J. Coyne, H.L. Feinstein, Youman, C.E. (1996), “Role-Based Access Control Models”, IEEE Computer, vol. 29 n. 2, pp. 38-47.

Sandhu, R.; Ferraiolo, D. and Kuhn, R. (2000), “The NIST Model for Role-Based Access Control: Towards a Unified Standard”, 5th ACM Workshop on Role-based Access Control, pp. 47-63.

Sundsted, T. (2002), “With Liberty and single sign-on for all. The Liberty Alliance Project seeks to solve the current online identity crisis”, Available http://www.javaworld.com/javaworld/jw-02-2002/jw-0215-liberty_p.html

Thompson, M. (1999), “Certificate-based Access Control for Widely Distributed Resources”, 8th USENIX Security Symposium, pp. 215-227.

W3C (2000), "SOAP: Simple Object Access Protocol 1.1". Available http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

W3C (2001), “Web Services Description Language 1.1”, Available http://www.w3c.org/TR/wsdl

W3C (2002), “Semantic Web Activity Statement”. Available http://www.w3.org/2001/sw/Activity

Yagüe, M.I. (2002), “On the suitability of existing access control and DRM languages for mobile policies”. Technical Report n LCC-ITI-2002-10, Department of Computer Science, University of Málaga. 2002.

Yagüe, M.I. (2003), “Modelo basado en Metadatos para la Integración Semántica en Entornos Distribuidos. Aplicación al Escenario de Control de Accesos”, Ph.D. dissertation, Computer Science Department, University of Málaga.