# Survey on XML-Based Policy Languages for Open Environments

Mariemma I. Yagüe

Computer Science Department,
University of Málaga. Spain
*yague@lcc.uma.es*

**Abstract:** This paper reviews several XML based languages and analyses their suitability for use in a mobile policy environment for access control and Digital Rights Management. A set of essential features is identified and a comparison between the different proposals is presented, along with some conclusions on the suitability of each language to the new applications emerging on the Internet.

***Keywords*:** XML Policy Languages, Use and Access Control, Mobile Policy, Distributed and Open Environments, Digital Rights Management**.**

## 1. Introduction

Since its inception, XML has been used for defining specific vocabularies to represent different human endeavors. In the context of policy specification, several XML based languages such as XACL[1], XrML[2], ODRL[3], SAML[4] or XACML[5] have been developed for access control, digital rights management, authentication and authorization. The eXtensible rights Markup Language (XrML) and extensible Access Control Markup Language (XACML) are two proposals for standard XML extensions in the fields of digital rights management and access control. Although each language has been designed for a specific problem, they share some common properties while others are specific to the environment they were designed for.

In this survey we focus on a distributed environment such as the Internet. In particular, we analyze different languages taking into account the access control to some new applications in this environment, such as Web services, grid computing, digital libraries, and electronic commerce systems. Some important features of these are:

- Their highly distributed nature;
- The heterogeneity of resources, access control requirements and users;
- The necessity of granting access to previously unknown or non registered users;
- The volatility of the user-resource relation;
- The dynamism of the Web. Resources are changed, eliminated and incorporated at any moment; moreover, access control requirements change frequently and the system must adapt to new conditions in a dynamic and transparent way;

Therefore, some requirements for access control in this scenario are:

- The generality of the solution, that must be applicable to different scenarios providing interoperability among the different systems;
- The scalability, since these environments present a high volume of heterogeneous resources from different organizations, along with numerous users and access control requirements;
- The ease of management of the access control system, because of the large number of users and resources, the dynamic adaptation to changes and the intrinsic difficulty for the correct definition of access control criteria.
- The distributed management of the access control disregarding resource physical location. This property is necessary because of the existence of different parts concerned with the access control.
- The distributed enforcement of the access control, avoiding the bottlenecks associated with centralized access control;
- The security level that must be comparable to centralized systems.
- Originator retained control. Originators should be able to retain control on the resources they own even after access is granted to users.
- Distributed access control management. Administrators should be able to manage the resources they control regardless of the location of that resource.
- Distributed access control enforcement. Access control mechanisms must be distributed in order to avoid bottlenecks in request processing.
- Flexibility. The system should be applicable in different scenarios.
- Independence. The system should not depend on underlying infrastructures or authentication systems.
- Dynamism. There should be a fast and secure mechanism to change policies.
- Ease of management. The distributed approach should not introduce complexity of management.
- Efficiency. Both access control management and enforcement should be efficient.
- Security. The distributed access control mechanism must ensure the same level of security as a centralized one can achieve. Tools to help security administrators should be provided.

Taking these features into account, we study the different proposals for access control and digital rights management, with the aim of defining the minimum set of features necessary to satisfy the requirements posed by these environments.

To solve some of the limitations of RBAC systems (Role Based Access Control) [6] an interesting approach based on the concept of mobile policies [7] has recently been

Received December 27, 2005. 

proposed. This proposal addresses the remote execution of access control policies, solving some of the problems found in centralized access control. The requirement of mobile policies for access control to be executed in trusted data servers, limits its practical usefulness. Moreover, because the mobile policy is integrated with the data that it controls, any change in the access requirements (policy) requires the data-policy package to be rebuilt and distributed again to each trustworthy data server.

A variant of this system, where the execution of policies in non trusted systems is enabled and the dynamic and transparent modification of policies is made possible, is presented in [8]. In this approach policies are not integrated but are simply linked to the object. Our access control mechanism will be based on this approach. Consequently, in this report we will consider the applicability of the different languages to this model. Among the specific features of mobile policies we have to mention the need for them to be very compact or, at least, to be easily translated into a compact format, and also the efficiency of the policy evaluations.

## 2. Analysis of Proposals

In this section the more relevant features of different access control and digital rights management languages based on XML are described and analyzed. These languages have been selected on the basis of appropriateness, relevance and maturity level.

### 2.1 Author-X

The Author-X [9][10] system was designed to provide access control for XML documents. The design approach makes it only useful for XML documents and their DTDs (in the case to be defined) stored in an XML database such as eXcelon. Both the sources to be protected and the authorizations of the system are stored in XML format. The main drawback is that objects to be protected are limited to XML documents and they must be stored in a local database. Also, the language for the specification of policies is based on DTDs, which have certain limitations unlike other more advanced languages such as XML-Schema. The language for the definition of access policies is based on credentials issued by the access control manager. Therefore, in practice, each credential will be useful just for a source, limiting interoperability. As a consequence users are obliged to subscribe to sources before they can access their contents. The user credentials are assigned at the registration phase. This approach is not well suited for the environments considered in this report, where avoiding the local registration is considered a basic requirement.

Furthermore, this scheme does not work well for scenarios where heterogeneous contents are frequent and the structure of user groups can not be anticipated by the administrators.

This system defines a hierarchic access control scheme based on the structure of the document. But the structuring of XML documents does not necessarily match the security requirements of the nodes. As a consequence, for the general case, the number of different authorizations (positive and negative) that have to be defined grows rapidly. The use of positive and negative authorizations increases the complexity of the access control system and also the chance of introducing errors in the specification of policies.

Author-X represents essentially a centralized scheme, although a distributed approach is proposed based on a set of federated sources relying on a central ‘master source’. The design based on this central ‘master source’ has negative consequences on its scalability.

The content protection of Author-X is founded on the concept of “passive” secure container requiring a different key for each possible view of the document. This results in serious disadvantages related to the administration of the access control system and the security [10].

### 2.2 FASTER

The FASTER project [11][12] and the Author-X system share some features such as the definition of a hierarchical access control schema based on the structure of the document, the restriction to XML sources and the specification of positive and negative authorizations.

In our environment it is normal that new resources are incorporated into the system frequently; and that each resource needs a different structure of group and access control policy. Therefore, in the fixed hierarchy used by FASTER the representation of the security requirements of the different contents to be protected is extremely complicated. Moreover, these security requirements for each resource may change over time. Another drawback is that the system is completely server-side.

FASTER does not support any content protection mechanism, except the creation of the appropriate user view on the server by a document pruning process. Some of its authors are now working on the development of the XACML language, presented in section 2.7.

### 2.3 XrML, eXtensible rights Markup Language

XrML is a digital rights specification language proposed by ContentGuard. Although it is a mature specification, its complexity and specificity makes it inadequate for scenarios where specifications must be kept simple. Since precision is one of its basic design objectives, it becomes inflexible. XrML is focused on the expression of rights for digital resources. It enables the specification of the trust level required for concerned parts, supports the identification of resources through UDDI (Universal Description, Discovery and Integration), and is supported by the use of XPath and digital signatures. The main concepts of XrML are license, grant, principal, right, resource and condition. These concepts can be extended to apply XrML to specific business environments.

The high level of detail regarding security used in the specification, limits the application of this language. The model is based on the user identity, an approach that does not fill in well with several scenarios such as those considered in this report, where it is neither adequate nor necessary that users be registered and identified.

XrML has been criticized because its development process has not been public and the design criteria are not known.

### 2.4 ODRL, Open Digital Rights Language

The ODRL language was submitted to the ISO/IEC MPEG group as a proposal for a rights language in the MPEG-21 framework. ODRL is based on XML-Schema, but no management tool is provided.

Digital rights management (DRM) is concerned with the description, analysis and evaluation of the rights possessed by rights-holders upon tangible or intangible objects. ODRL provides the semantics for expressions related to DRM in open and trusted environments, although it does not take into account support to the security mechanisms.

ODRL provides the digital version of the traditional rights management systems. It also supports an extensible range of new services which appear in the digital and Web environment. This standard language defines a vocabulary for the expression of terms and conditions on digital and physical objects. Among other things, ODRL allows users to define who possess the rights, the kind of uses allowed, as well as the offers and contracts relating to these objects. The capacities and requirements of the underlying systems with respect to the content protection, physical or digital distribution and payment are not indicated in the ODRL specification.

The ODRL specification includes a basic model, which allows the description of objects, rights and parties. Models for permissions, restrictions, requirements, conditions, rights holders, context, offer, contract, revocation and security are also defined.

In summary, the objectives of this language differ from those considered in this report. Nevertheless some elements could turn out to be useful from our point of view. These are permissions and requirements (see sections 2.2 and 2.4 of the specification).

### 2.5 XACL, XML Access Control Language

XACL was one of the first XML-based proposals for a language to specify access policies. The language, developed by IBM, follows the traditional model of control access systems based on four concepts:

- Object: resource to control the access. The granularity of this object is at document element level.
- Subject: entity, which request the access. This concept includes identity, group and role.
- Action: that is the subject is trying to carry out on the object. This specification just defines four possibilities (read, write, create and delete) although the structure of the language does not limit the addition of others.
- Condition: that must be fulfilled for the subject to make the action over the object.

The proposed architecture is specific to controlling the access to XML documents. It is difficult to adapt this scheme to other environments. The language specification is based on DTDs, and is therefore limited by their lack of expressiveness. The notion of subject comprises identity, group, and role. The granularity of the model is as fine as single elements within the document. Currently, there are four possible actions (read, write, create and delete), but the structure of the language is not limited to these.

XACL supports a provisional authorization model [13], where users can specify provisional actions associated with a primitive action (read, write, create, or delete). Almost all studies in access control and authorization systems have assumed the following model: "a user makes an access request of a system in some context, and the system either authorizes the access request or denies it." In the provisional authorization model, the answer from the system is not simply "grant" or "deny." It tells the user that his request will be authorized provided he (and/or the system) takes certain actions or that his request is denied but the system must still take certain actions. Such actions are called provisional actions. Examples of provisional actions include auditing, digital signature verification, encryption, and XSL transformations in addition to write, create and delete actions.

The architecture of the authorization model comprises two main modules: the first one for the evaluation of the access decision and the second one for the execution of the request. The second module is necessary to guarantee that provisional actions are taken. The XACL specification establishes that the authentication of the subject and the role assignment are out of its scope.

As a conclusion of the study of this proposal we have to mention the benefits of the provisional authorization model, which enables more expressive and flexible access control systems. On the other hand the model based on a centralized execution of policies is much more restrictive, and it is not adaptable to the environments considered in this report which present a high level of distributed execution, and where the centralized model would represent a serious disadvantage. Additionally its specificity, which is very oriented to controlling the access to XML documents, limits its use.

### 2.6 SAML, Security Assertion Markup Language

SAML s a proposal of OASIS (Organization for the Advancement of Structured Information Standards) which includes an assertion language and a security messaging protocol to describe, request and send authentication and authorization data among security domains. Its main objective is to promote interoperability between different security systems representing an XML-based framework for electronic business transactions (e-business). One major design goal for SAML is for it to be used in Single Sign-On (SSO), i.e. where a user is able to authenticate in one domain and use resources in other domains without re-authenticating. Therefore, this objective differs from ours although some of its features are interesting.

An assertion is a package of information that supplies one or more statements made by an issuer. SAML allows issuers to make three different kinds of assertion statements.

- Authentication: The specified subject was authenticated by a particular means at a particular time.
- Authorization Decision: A request to allow the specified subject to access the specified resource has been granted or denied.
- Attribute: The specified subject is associated with the supplied attributes.

Assertions have a nested structure. A series of inner elements representing authentication statements, authorization decision statements, and attribute statements

contain the specifics, while an outer generic assertion element provides information that is common to all of the statements.

SAML allows the definition of the following elements: subjects, assertions (access conditions), authentication, authorization and attributes. A SAML expression consists of a series of assertions about a subject. A subject is an entity (either human or computer) that has an identity in some security domain. A typical example of a subject is a person, identified by his or her email address in a particular Internet DNS domain. Assertions can convey information about authentication acts performed by subjects, attributes of subjects, and authorization decisions about whether subjects are allowed to access certain resources. Assertions are represented as XML constructs and have a nested structure, whereby a single assertion might contain several different internal statements about authentication, authorization, and attributes. Note that assertions containing authentication statements merely describe acts of authentication that happened previously. Assertions are issued by SAML authorities, namely, authentication authorities, attribute authorities, and policy decision points. SAML defines a protocol by which clients can request assertions from SAML authorities and get a response from them. This protocol, consisting of XML-based request and response message formats, can be bound to many different underlying communications and transport protocols; SAML currently defines one binding, to SOAP over HTTP.

In summary, SAML supports some definitions that can be useful in our application scenario, such as the Conditions, Action, Attribute, and AuthorizationDecisionStatement elements (see section 2 of the specification).

### 2.7 XACML, eXtensible Access Control Markup Language

XACML represents an initiative of OASIS for the definition of a language to express access policies to information identifiable by XML on the Internet. XACML is expected to address fine grained control of authorized activities, the effect of characteristics of the access requestor, the protocol over which the request is made, authorization based on classes of activities, and content introspection (i.e. authorization based on both the requestor and also many attribute values within the target. The values of the attributes may not be known to the policy writer). XACML is also expected to suggest a policy authorization model to guide implementers of the authorization mechanism.

The syntax of the policy specification is based on a {subject, object, action} tuple. The subject element can express user-Ids, groups and roles. The object element enables a fine granularity (elements inside an XML document). Finally, the action element consists of four different kinds of actions: read; write; create and delete. The concept of provisional authorization has also been included. Some criticisms have been made to the technical committee regarding certain decisions in the specification approach. Among the more important concerns is the excessive bias towards the control of XML documents. The definition based on tuples of XACML is simple and powerful for some environments. For other scenarios its results are limited in, at least, the following three aspects:

Firstly, the assumption, included in the design of the language, that every object to be protected is going to be an XML document or a part of it, is obviously unfortunate and is not appropriate for environments where the objects to be protected are other types of documents, data, devices, etc. Furthermore, the direct association between objects and subjects has a limited value. What is really needed is the possibility to define relations in terms of user and object properties. The fact that XACML enables a role name to be used as the subject is a step in this direction, because a role is just a name for a privilege set. However, this is not general enough because it does not allow the allocation of privileges to users in a direct way (it can only be done through their roles). Moreover, the concept of 'object properties' has not been considered at all. Finally, the explicit inclusion of the action element (representing the action to be carried out on the object) in the policy language is a strong limitation to the predefined actions (read; write; create; and delete). In any real scenario, in particular in those considered in this report, the number of possible actions is unlimited. The first objective of the standardization is to guarantee interoperability. Therefore, the definition of an incomplete set of actions is not a good option. Instead, the provision of a means to specify any action is a better approach regarding interoperability.

The context and schema of XACML are described in three models that elaborate different aspects of its operation. These models are: the data-flow model, the policy language model and the administrative model.

The major actors in the XACML domain are shown in the data-flow diagram of Figure 1.

Policy administration point (PAP) – The system entity that creates a policy or policy set

Policy decision point (PDP) – The system entity that evaluates applicable policy and renders an authorization decision

Policy enforcement point (PEP) – The system entity that performs access control, by enforcing authorization decisions

Policy information point (PIP) – The system entity that acts as a source of attribute values

Policy mediation point (PMP) – The system entity that resolves policy conflicts

Policy retrieval point (PRP) – The system entity that locates and retrieves applicable policy for a particular decision request

Some of the data-flows shown in the diagram may be facilitated by a repository. For instance, the communications between the PDP and the PIP or the communications between the PDP and the PRP or the communication between the PAP and the PRP may be facilitated by a repository. The XACML specification is not intended to place restrictions on the location of any such repository, or indeed to prescribe a particular communication protocol for any of the data-flows.
The model operates according to the following steps.

1. PAPs write policies and make them available to the PRP. From the point of view of an individual PAP, its policies represent the complete policy for a particular target. However, the PDP may be aware of other PAPs that it considers

authoritative for the same target. In this case, it is the PDP's job to obtain all the policies and combine them in accordance with a policy-combining algorithm. The result should be a self-consistent policy set.

2. The PEP sends an authorization decision request to the PDP, in the form of a SAML request. The decision request contains some or all of the attributes required by the PDP to render an authorization decision, in accordance with applicable policy.
3. The PDP locates and retrieves the policy applicable to the decision request from the PRP.
4. The PRP returns the applicable policy to the PDP in the form of an XACML <PolicyStatement> or <PolicySetStatement>. The PDP ensures that the decision request is within the scope of the <PolicyStatement> or <PolicySetStatement>.
5. The PDP examines the authorization decision request and the policy to ascertain whether it has all the attribute values required to render an authorization decision. If it does not, then it requests attributes from suitable PIPs in the form of SAML requests of the attribute query type.
6. The PIP (which may be a SAML attribute authority) locates and retrieves the requested attributes from other systems by some means, and in a form, that is out of the scope of this specification.
7. The PIP returns the requested attributes to the PDP in the form of SAML responses containing SAML attribute assertions. The PDP evaluates the policy.
8. If the policy were to be evaluated TRUE, then the PDP returns an authorization decision, in the form of a SAML response, to the PEP containing the "Permit" saml:Decision attribute and (optional) obligations.
9. The PEP fulfils the obligations.

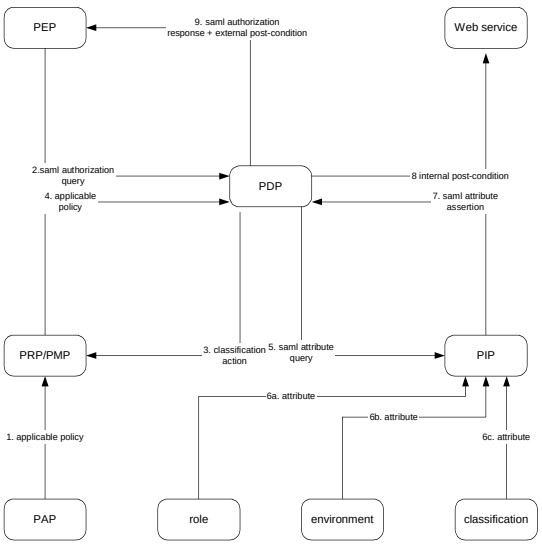


**Figure 1.** Data-flow diagram of XACML

### 2.7.2 Policy Language Model

The policy language model is shown in figure 2. The model is divided into six parts.

The section principal/role/attribute of the language model includes the principal, role, role attribute and attribute classes. An authorization request is related to a single principal. XACML policy instances may reference attributes of a particular principal, or a role of the principal. The PDP should use attribute assertions to confirm whether the principal occupies a role specified in policy. Both the principal and the role may have attributes. For instance, the principal "Joe" may have an attribute of type "role" set equal to the value "purchasing officer". Alternatively, the role "purchasing officer" may have an attribute of type "signing limit" set equal to the value "100,000€". Principal and role attributes are asserted by authorities and distributed in the form of SAML attribute assertions. The PDP is responsible for checking that the attribute values it operates upon are asserted by suitable authorities.

The resource/classification/attribute section of the language model includes the resource, classification, classification attribute and attribute classes. An authorization request is related to a single resource. XACML policies may reference attributes of a particular resource or a classification of the resource. The PDP is also responsible for confirming that the resource occupies the required classification and for locating and retrieving the resource attributes referenced by the applicable XACML policy instance. The PDP is also responsible for checking that suitable authorities assert the attribute values it operates upon. In the case where the resource is an XML document, the resource classification may be an attribute or element within the resource itself. In other cases, resource and classification attributes may be asserted by authorities and distributed in the form of SAML attribute assertions.

Both the resource and classification may have attributes. For instance, a purchase order may have an attribute of type "total price" set equal to the value "87,750.00€". Alternatively, the classification "capital equipment" may have an attribute of type "category of goods" set equal to the value "computer equipment".

The environment/attribute section of the language model includes the environment attribute and attributes classes. XACML policy instances may reference attributes that are not directly associated either with the principal or the resource. These attributes are called environment attributes. For instance, the "current time of day" is an environment attribute that may be referenced by a policy instance. Environment attributes are asserted by authorities and distributed in the form of SAML attribute assertions. The PDP must check that suitable authorities assert the attribute values it operates upon.

The target/action/classification section of the language model includes the resource, target, classification and action classes. Policy instances are identified with a classification/action pair. The PDP is responsible for checking that the policy instance used to compute the authorization decision is applicable to the authorization request. It does this by verifying that the action identified in the authorization request is the same as the action identified

in the policy instance, and that the resource identified in the authorization request belongs to the classification identified in the policy instance. The algorithm for matching a resource name to a classification name is identified by a URI. A regular expression may be used for resources in the URI namespace.

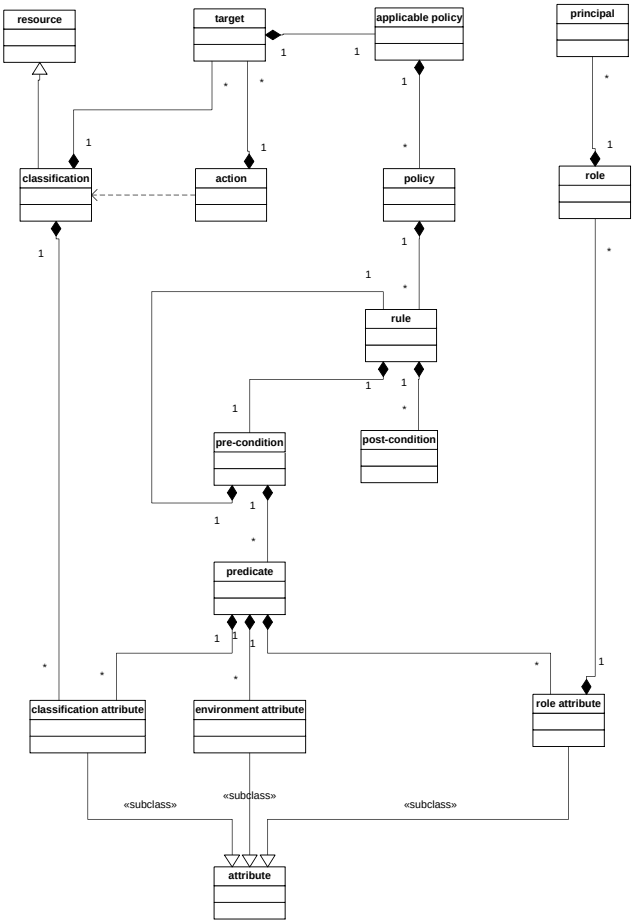


**Figure 2**. XACML Policy Language Model

The policy/rule/pre-condition/predicate section of the language model includes the policy, rule, pre-condition y predicate classes.

XACML policy instances are built from a logical combination of rules. Each rule comprises one pre-condition and zero or more post-conditions. A pre-condition is a logical operator or predicate. A predicate is a statement about attributes that can be verified by the PDP. If the policy instance applicable to an authorization request is evaluated TRUE, and all internal post-conditions are satisfied, then the PDP may return an authorization decision attribute with the value "permit" to the PEP.

The post-condition section of the language model only includes the post-condition class. Post-conditions are actions specified in an XACML policy instance. Post-conditions are of two types. Internal post-conditions must be successfully executed prior to returning an authorization decision attribute with the value "permit". External post-conditions must be returned by the PDP to the PEP and an authorization decision attribute with the value "permit" may be issued without confirmation that the condition has been successfully executed.

2.7.3Administrative Model

It is essential that XACML policy instances only contain references to attributes and post-conditions that are accessible by the PDP or PEP. The administrative model, shown in figure 3, illustrates how this is achieved. The various SAML attribute authorities involved must provide an interface by which the policy administration point can discover the attribute types available from it.

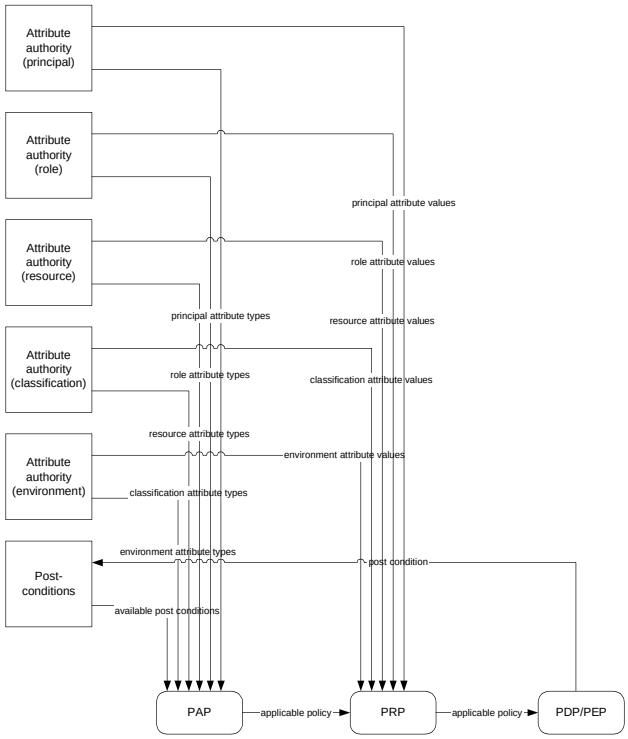


**Figure 3.** Administrative Model of XACML

The core scheme of XACML is extensible to new features that can be needed. XACML is based on an XML Schema for representing authorization policies and entitlement. However, it must be noticed that a completely different representation for the PDP can be selected for its internal evaluation and its decision making process. That is, for XACML it is admissible that some systems treat it just as a format for the interchange of policies, and in this way some implementations will translate the XACML policy to their own proprietary language or alternative before the evaluation. Every PDP input and output must be SAML compatible (in accordance with the XML-based Secure Content Distribution (XSCD) infrastructure, which is based on the production of protected software objects that convey contents (software or data) and can be distributed without further security measures because they embed the access control enforcement mechanism. It also provides means for integrating Privilege Management Infrastructures (PMIs). assertions and messages of protocol defined in the SS-TC SAML specification) although they can support other formats and syntax for the PDP input and output.

### 2.8 Semantic Policy Language

As we mentioned in section 1, 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 open and distributed scenarios. On the other hand, some important

features are not considered in these languages. For this reason, we developed a specific XML-based language to specify the access control policies [14]. This language was called Semantic Policy Language because it is based on semantic properties of the resources to be accessed, the external authorization entity (Privilege Management Infrastructure, PMI) and the context. These semantics are used during the specification of access control criteria, dynamic policy allocation, and 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, parameterization and abstraction. Additionally and as previously 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 SPL implies: 1. Separation of specification in three parts: access control criteria, allocation of policies to resources and semantic information (properties about resources and context). 2. Abstraction of access control components. 3. Ability to reuse these access control components. 4. Reduction of the complexity of management due to previously mentioned 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 parameterized allowing the definition of flexible and general policies, thus reducing the number of different policies to be managed. Parameters, which can refer to complex XML elements, are instantiated dynamically from semantic and contextual information.

Additionally, policies can be composed by 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 to reduce 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.

## 3. Summary of Features

From the study of the languages included in this survey we have selected the proposals that present features in order to make them useful for the targeted scenarios, and a set of issues has been studied. The following tables summarize the results comparing them with the actual features of the SPL language [17], developed by the author:

| LANGUAGE | | | | | |
|---|---|---|---|---|---|
| | X-Author | FASTER | XACL | XACML | SPL |
| Policy Specification Method | Credential-based. | Based on RBAC and hierarchies. | RBAC. | Based on identity and credentials. | Based on attribute certificates |
| Syntax | DTD | XML Schema | DTD | XML Schema | XML Schema |
| Complexity Level | Low | Medium | Low | High | Low |
| Expressiveness | Medium | Medium | Low | High | High |
| Ambiguity | Possible because of positive and negative authorizations | Possible because of positive and negative authorizations | Possible because of positive and negative authorizations | Possible. Requires the PMP to resolve conflicts | NO |
| Modular Language | NO | NO | NO | YES | YES. Modular policies and with parameters. They can also be composed without ambiguity. |
| Semantic Validation | NO | NO | NO | NO | YES. Automatic detection of inconsistencies and errors based on the semantic information about the context, the resource to be accessed and the authorization entities. |
| Content-based Access | NO. Based on Structure | NO. Based on Structure | NO. | Dependent on Implementation. | YES. At the semantic level |

| | | | | | |
|---|---|---|---|---|---|
| | (DTDs). | (XML Schemas). | | | (metadata). |
| Scalability | Low.<br>Based on subscriptions.<br>Credentials registered locally. | Medium.<br>Based on certificates but it requires subscription (for the identification). | Low.<br>Centralized. | Dependent on the Implementation. | High.<br>Fully distributed scheme and certificate based.<br>No subscription is required. |
| Interoperability Level | Low.<br>Federated Sources | Low | Null | Medium-High<br>(Not sufficiently specified)<br>Based on SAML assertions. | High.<br>Integration of Privilege Management Infrastructure based on metadata about the Source Of Authorizations. |
| Policies can be modified in a dynamic and transparent way | NO | YES (centralized) | YES (centralized) | NO | YES |
| SYSTEM | | | | | |
| | X-Author | FASTER | XACL | XACML | SPL |
| Dependency of the Language | YES | YES | YES | Dependent of the implementation | NO<br>It could use any language that becomes the standard. |
| Application Scope | XML documents (valid respect to a DTD or simply web formed) | XML documents | XML documents | Resources identifiable through a URI (anyURI) | Not restricted:<br>Software Objects (distributed objects, Web services, applets, servlets...)<br>Data Objects (without format restriction: multimedia objects, forms, XML, ...) |
| Integration with external authorization mechanisms | NO | Possible.<br>Not defined. | NO | Possible.<br>Uses SAML. | Complete.<br>X.509 Standard and semantic information. |
| APPROACH | | | | | |
| | X-Author | FASTER | XACL | XACML | SPL |
| Generality | Specific Purpose | Specific Purpose | Specific Purpose | Specific Purpose | General Purpose |
| Access Control Scheme | Identification.<br>Language based on DTDs to express credentials and its types. | Identification.<br>Language based on XML-Schema for the expression of identity and attribute certificates. | Identification.<br>Language based on DTDs to express RBAC elements (iduser, group, role). | Identification.<br>Language based on XML-Schema for the expression of identity and conditions about the attribute certificates, the resource and the environment. | Attributes.<br>Language based on XML-Schema for the specification of conditions related to the attribute certificates, the resource and the context.<br>Complemented with other semantic components of the language (PAS, SRR, SOADs) |
| SECURITY | | | | | |
| | X-Author | FASTER | XACL | XACML | SPL |
| Secure Distribution | YES<br>Passive Containers.<br>Problems with key Management. | NO | NO | NO | YES<br>Active Containers. |
| Distributed Mechanism for Policy Execution | NO<br>Centralized Execution | NO<br>Centralized Execution | NO<br>Centralized Execution | NO<br>Centralized Execution | YES<br>Based on Active Containers |
| Provisional Authorization | NO | NO | YES | YES | YES |
| Temporary Authorization | NO | NO | NO | NO | YES |
| PAYMENT | | | | | |
| | X-Author | FASTER | XACL | XACML | SPL |
| Supported by Language | NO | YES | NO | Possible<br>(not defined) | YES |
| Implemented | NO | NO | NO | NO | YES |

## 4. Conclusions

It must be emphasized that XML is the best alternative for defining policy languages. Among the features and application scenarios considered we can extract the following requirements for any language suitable to these environments:

- Simplicity
- Flexibility
- Expressiveness
- Modularity
- Scalability
- Interoperability
- Extensibility
- Lack of ambiguity
- Open access control scheme
- Integration with external authorization schemes
- Access based on contents (content introspection)
- Integrated Solution (administration, enforcement)
- Provisional Authorization (in particular, payment methods)
- Temporary Authorization
- Distributed Execution of policies
- Mechanisms for Secure Distribution of contents

With respect to the criteria considered in section 3 we can conclude that SPL (Semantic Policy Language) presents a set of features that makes it very suitable for the application scenario objective of this report. In fact, some of the more relevant features are excusive to this proposal. To the best of our knowledge no other work incorporates them. It is important to mention that the comparison with XACML, considered the best candidate, presents important advantages in favor of SPL. Because of the open nature of SPL, the features where XACML overcomes SPL can be easily included in the latter.

In our opinion, the advantages presented by SPL are directly derived from a series of basic design criteria, such as the extensive use of semantic information, the structure of the language composed of different parts (policies, PAS, SRR, SOADs, contextual information, etc.), the integration with an external authorization structure, and finally, the development of an integrated environment that incorporates administration tools for the creation and validation (syntactic, semantic and contextual) of policies [15-17]. Additionally, a complete infrastructure was developed called XSCD, which combines the Semantic Access Control Model (with the SPL and the semantic integration of the PMI) and a software protection mechanism called *SmartProt* [18] in order to provide distributed access control management and enforcement, and secure content distribution in open environments. XSCD [19-20]. In the XSCD system policies can be dynamically changed by the owner or originator of the resource in a transparent manner.

## References

[1] Kudo, M., Hada, S. "XML Document Security based on Provisional Authorisation". In *Proceedings of the 7th ACM Conference on Computer and Communications Security*. 2000.

[2] ContentGuard, Inc. "eXtensible Rights Markup Language, XrML 2.0". 2001. http://www.xrml.org

[3] Open Digital Rights Language Initiative. "Open Digital Rights Language". http://odrl.net/. 2001.

[4] Organization for the Advancement of Structured Information Standards. "Security Assertion Markup Language". http://www.oasis-open.org/committees/security/ . 2002

[5] Organization for the Advancement of Structured Information Standards. "eXtensible Access Control Markup Language". http://www.oasis-open.org/committees/xacml/

[6] Sandhu, R.S., E.J. Coyne, H.L. Feinstein and Youman, C.E., "Role-Based Access Control Models", *IEEE Computer*. 29(2): pp. 38-47. 1996.

[7] Fayad, A., Jajodia, S. "Going Beyond MAC and DAC Using Mobile Policies". In *Proceedings of IFIP SEC'01*. Kluwer Academic Publishers. 2001.

[8] López, J., Maña, A., Yagüe, M.I. "XML-based Distributed Access Control System". In *Proceedings of 3rd Int. Conference on Electronic Commerce and Web Technologies*. Lecture Notes in Computer Sciences vol. 2455. 2002.

[9] Bertino, E., Castano, S., Ferrari, E. "On Specifying Security Policies for Web Documents with an XML-based Language". In *Proceedings of ACM SACMAT'01*. 2001.

[10] Bertino, E., Castano, S., Ferrari, E. "Securing XML documents with Author-X". *IEEE Internet Computing*, 5(3):21-31, May/June 2001.

[11] Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P. "Controlling Access to XML Documents". In *IEEE Internet Computing*, vol. 5, n. 6, November/December 2001, pp. 18-28.

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

[13] S. Jajodia, M. Kudo, and V. S. Subrahmanian, "Provisional Authorizations", In *Proceedings of the Workshop on Security and Privacy in E-commerce* (WSPEC). 2000.

[14] Yagüe, M.I., Troya, J.M. "A Semantic Approach for Access Control in Web Services". In *Proceedings of Euroweb 2002 International Conference*. W3C and British Computer Society Electronic Workshops in Computing (eWiC). 2002.

[15] Yagüe, M.I. "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. 2003.

[16] Yagüe, M.I., Gallardo, M., Maña, "A. Semantic Access Control Model: A Formal Specification". In *Proceedings of 10th European Symposium On Research In Computer Security, ESORICS*. LNCS 3679, Computer Security- ESORICS 2005, pp. 24-43. 2005.

[17] Yagüe, M.I., Maña, A., López, J., "A Metadata-based Access Control Model for Web Services''. *Internet Research Journal: Electronic Networking Applications and Policy*, 25(1), 99-116. 2005.

Received December 27, 2005. 

[18] Maña, A. and Pimentel, E., "An Efficient Software Protection Scheme", in *Proceedings of IFIP SEC'01*. Kluwer Academic Publishers. 2001.

[19] Yagüe, M.I., Maña, A., López, J., Pimentel, E., Troya, J.M. "A Secure Solution for Commercial Digital Libraries". *Information Review Journal*, 27(3), 2003, 147-159. Emerald Publishers. 2003.

[20] Yagüe, M.I., Maña, A., López, J., Pimentel, E., Troya, J.M. "Access Control Infrastructure for Digital Objects". In *Proceedings of 4th. Int. Conference On Information and Communications Security 2002*. LNCS 2513. 2002.

## Author Biography

**Mariemma I. Yagüe** received her BSc and MSc degrees in Computer Engineering from the University of Granada (Spain) in 1990 and 1992, respectively. After working at the University of Vigo from 1993 until 1996, she joined the Department of Computer Science at the University of Málaga (Spain) in 1996 where she received her PhD degree in Computer Science.

Her main research activities are primarily in the area of metadata and security on distributed information systems. More specifically, access control, authorization, digital rights management (DRM) and semantic modeling. The application of semantic information to the field of access control resulted on the development of a new access control scheme, the Semantic Access Control Model (SAC) as her Ph.D. work. She is very involved in several national and international research projects and NoEs, and is member of the CEN/ISSS Digital Rights Management Working Group, the Open Digital Rights Language Working Group, the CRIPTORED Ibero-American Thematic network on Cryptography and Information Security, among others. She participates in the organization of different events, including the First International Workshop on Technological & Security Issues in Digital Rights Management (in conjunction with ESORICS'06), the Fourth International Workshop on Security In Information Systems (WOSIS-2006), the Training Workshop of Security and legal aspects of Digital Rights Management at Smart University'06, , the Sixth European Intensive Program on Information and Communication Security Technologies, the USB Smart Card Summer University School, the Sixth International Conference on Information and Communications Security, the CNIS Special session on "Architectures and Languages for Digital Rights Management and Access Control", etc.

She participated in the development of the EC-gate system, a digital rights management tool that received the Gold Award of the e-gate Open 2002, celebrated in Paris and organized by Axalto, Sun Microsystems and ST Microelectronics.