Seventh LACCEI Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2009)
"Energy and Technology for the Americas: Education, Innovation, Technology and Practice"
June 2-5, 2009, San Cristóbal, Venezuela

A Project on Web Services Security and Reliability

Eduardo B. Fernandez
Florida Atlantic University, Boca Raton, Florida, USA, ed@cse.fau.edu

María M. Larrondo Petrie
Florida Atlantic University, Boca Raton, Florida, USA, petrie@fau.edu

ABSTRACT

We completed a one-year project funded by the U.S. Dept. of Defense to evaluate the state of the art of web services security and reliability. There are numerous standards for security and reliability and we analyzed their possible value for system design. In any large system we need to incorporate a good number of COTS components and we also considered what relevant products are available and what standards they support. Tools that can help designers build this type of systems were another objective. Finally, we studied how to use all this in a systematic lifecycle-based methodology to build secure and reliable systems. A byproduct was the development of several architectural patterns to be used in such a methodology to help designers incorporate standards in products and evaluate existing products. The project included three faculty members and two MS-level students. Several publications and two MS theses resulted from this work.

Keywords: Web Services, Security, Reliability, Software Engineering

1. INTRODUCTION

Web services are software components defined by their interfaces that can be accessed on the Internet and incorporated into applications. Web services communicate using EXtensible Markup Language (XML) messages that typically follow the Simple Object Access Protocol (SOAP) standard. They are becoming more and more the fundamental building blocks of distributed systems. Their value comes from their increased use in commercial systems and the fact that they are the basic building blocks of computational grids such as the Global Information Grid. Web services are a realization of a more abstract architectural style called Service-Oriented Architecture (SOA).

While web services introduce a variety of new useful functions, they increase the complexity of the system where they are used. Because of this complexity, web services can be subject to a variety of attacks. The fact that web services are being used for many sensitive areas, e.g. financial and military applications, provides a strong motivation for attackers. Web services security relies on *authentication* (verifying a user's identity based on submitted credentials), *authorization* (granting access to specific resources based on an authenticated user's rights), and *audit* (a log of activities). We completed a one-year project funded by the U.S. Dept. of Defense to evaluate the state of the art of web services security and reliability.

There is a large variety of standards for web services. This variety is bewildering for product developers and users. One of the reasons for the proliferation of standards is the number of organizations involved in developing them. We enumerated them to guide designers and users. We summarized this study in a set of tables.

A cornerstone of any defense to threats is a secure software architecture. We have been working on a secure system development methodology for the last three years [1]. This methodology is based on the use of patterns. We studied how this methodology fits into this approach.

The project surveyed the context for web services security, considered appropriate architectural levels, as well as issues and standards at each level. We try to evaluate the status of industrial practice with respect to the security of web services. We examine the relevant levels one by one and consider their security. We looked at commercial products and supporting levels. Reliability is another important aspect and we discussed some of its issues and considered its effect on security.

We believe in the value of patterns to build architectures and we provide an overview of the patterns used in SOA. A pattern is a reusable solution to a recurrent systems problem and their use has increased consistently in software development, being now adopted by any vendors and developers. We have proposed the idea of expressing standards as patterns and use these patterns to understand and compare the standards [2]. These patterns are also useful to evaluate existing products by checking if they include specific patterns. Clearly, for this idea to be useful we need patterns for the corresponding standards. We had produced earlier patterns for some of them and last year we have added seven more.

As an important practical aspect of this study we have kept track of products in the market that support web services security and we built a catalog of them. While not comprehensive, this catalog gives a good idea of what is available. We have used these product descriptions in the past to develop security patterns, we use them now to provide a study and analysis of the current state of these products and tools, including not just security but development and composition aspects.

In order to use patterns effectively we need catalogs. To organize catalogs we need a good classification of patterns. Part of our work in this project was dedicated to defining such a classification. We produced a multidimensional classification [3], and we are extending it in collaboration with a group at the National Institute of Informatics of Japan.

Validation and Certification approaches are important to improve trust in any product included in the global architecture and we describe some of the existing approaches. Governance is another new concept for web services, also briefly considered here.

2. WEB SERVICES SECURITY AND STANDARDS

Figure 1 summarizes the main general standards that apply to each layer, while Figure 2 shows most of the security standards.

There are also many fault tolerance mechanism which can be adopted in the design of a web service to increase reliability [4]. Replication and redundancy and diversity are a few mechanisms that can be used in the design of web services. In more detail N-version programming and recovery block applies diversity by having several different implementations of software or hardware specifications, running in parallel to cope with errors or failures that could arise directly from a specific implementation or design, in this case in a particular web service. Triple modular redundancy and dual modular redundancy mechanisms are used to duplicate critical web services in the event of failure, service is still available to consumers.

A recent standard for the workflow level is BPEL (Business Process Execution Language), developed by IBM, BEA, and Microsoft [5]. BPEL provides a language for the formal specification of business processes and business interaction protocols (called as a whole “web services choreography”). This extends the Web services interaction model and allows web services to perform business transactions.

The Security Considerations for the BPEL, briefly mentioned at the end of the specification, strongly recommend that business process implementations use WS-Security (see below) to ensure there have not been any modifications of messages during transit or while residing in destinations. WS-Security is a broad description of a framework indicating how to secure Web services including a family of WS-* standards and specifications including those related to federation such as WS-Federation, WS-Authorization, and WS-SecureConversation.

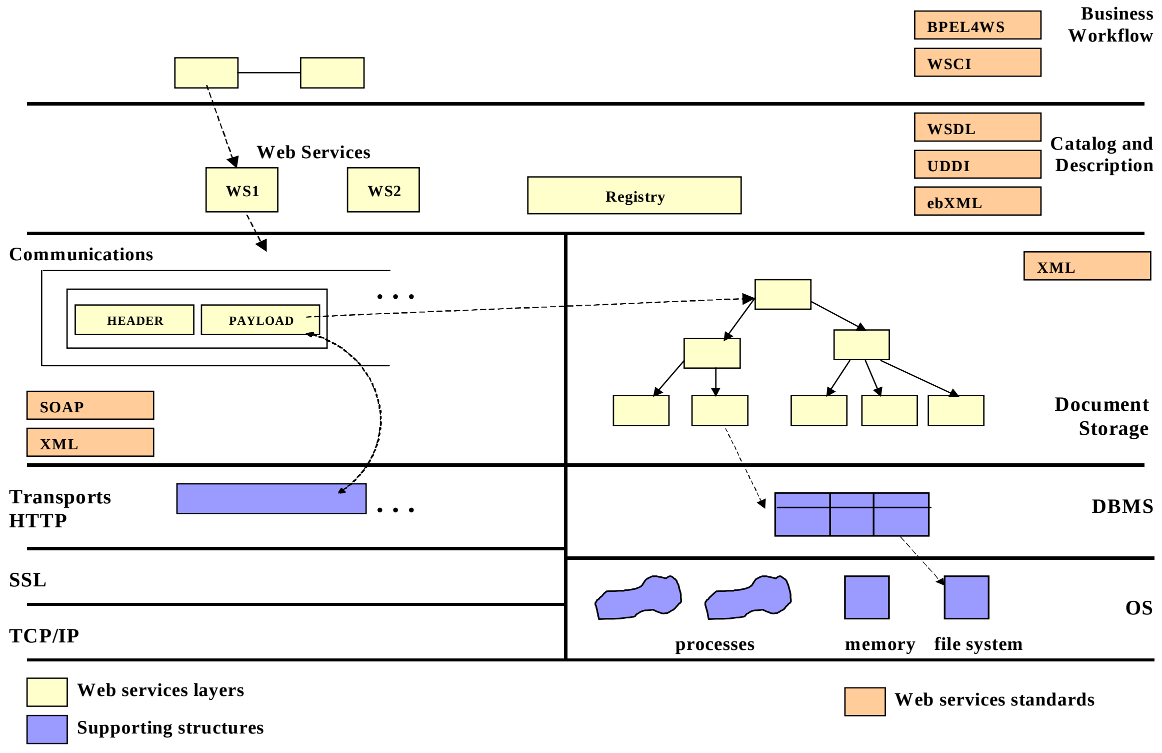

Figure 1. Layers and web services standards

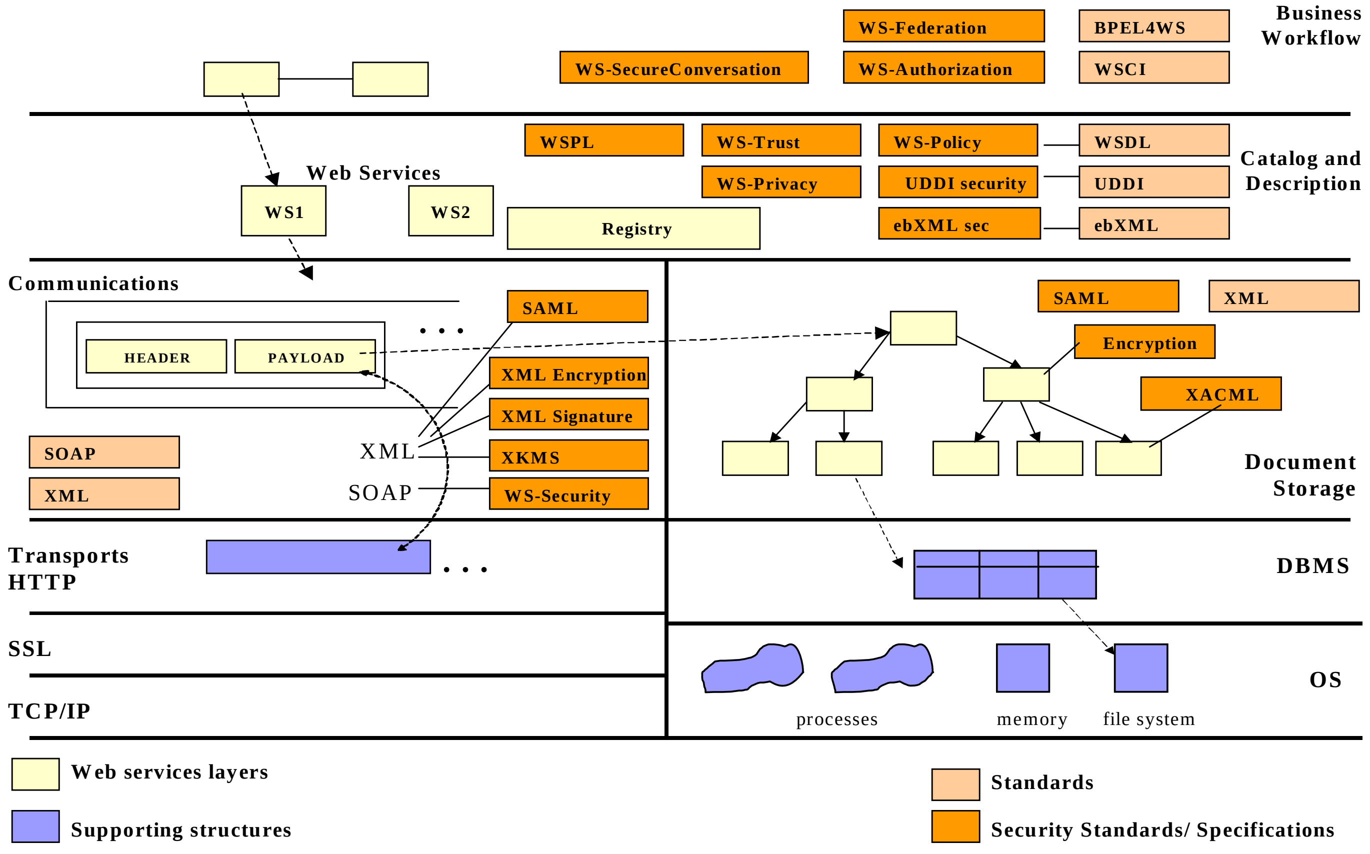

Figure 2. SOA architectural layers and web services security standards

Web services descriptions are defined using the Web Service Description Language (WSDL), another standard. WSDL is a static interface definition [6].

There are two primary e-business registries providing registration, management, and discovery information needed to conduct e-business, the Universal Description, Discovery and Integration (UDDI) specification and ebXML registry.

The UDDI directory specifies a metadata aggregation service. These metadata aggregation services are useful repositories in which organizations can publish services they provide, describe the interfaces to their services and enable domain-specific taxonomies of services.

The paucity of security specifications for UDDI and WSDL indicates a misunderstanding of their importance for security. The exposure of too many details could help hackers and their lack of integrity would severely affect the use of web services.

The Simple Object Access Protocol (SOAP) defines the following level. Actually there are other approaches that do not use SOAP, e.g., Representational State Transfer (REST), but they have not gained general acceptance.

SOAP security is defined by the SOAP Security Extensions. These extensions have been defined by the W3C XML Encryption Working Group [7]. One of these standards, XML Encryption, defines a process for encrypting and decrypting messages considering the granularity of the message contents. This can be as small as one element (including start/end tags) or apply to the element content (between the start/end tags). Super-encryption is possible, where the whole message with parts encrypted can be encrypted again, as is done when Secure Sockets Layer (SSL) is used for secure transmission over HTTP. A variety of encryption algorithms can be used, including the Advanced Encryption Standard (AES). Another standard from W3C for SOAP also specifies digital signatures (XML Signature). A SOAP message includes a header and a payload. Security Assertion Markup Language (SAML) defines authentication and authorization assertions. SAML assertions can be included in the header or in the payload of a SOAP message.

XML Encryption protects the secrecy of the message. A Public Key Infrastructure (PKI) can be used to provide authentication, digital signatures, and key distribution. This PKI could be based on XML Key Management Specification (XKMS), intended for the integration of PKI and digital certificates. For example, digital signature processing can be delegated to a web service in order to simplify the PKI structure. XKMS is an open standard that applies to any vendor PKI approach.

There are two standards for reliable messaging: WS-Reliable Message and WS-Reliability. Both have similar objectives, patterns for them and a comparison are given in [8].

One can (and should) use domain-based security according to document contents. There is already a good amount of research on XML security and a standard, XML Access Control Markup Language (XACML) is a language that can define authorization rules for each element of an XML document or for whole documents [9]. XACML is intended to provide a way to express access control policies in a variety of environments, using a single language. A rule has a subject (requesting entity), a right (read, write, etc.), an object (the document element), and a condition (for example, day of the week when access is permitted). XACML also describes an enforcement mechanism, where a request is evaluated and a decision is returned. In other words, XACML is an expression of an extended access matrix together with an enforcement mechanism.

3. SECURE SYSTEMS DEVELOPMENT METHODOLOGIES

Most of the approaches to produce secure software are based on analyzing code. While this is a reasonable approach, it will not have a strong impact in future systems. We believe that we need to emphasize the modeling aspects of code development and we have proposed a methodology for this purpose. A main idea in the proposed methodology is that security principles should be applied at every stage of the software lifecycle and that each stage can be tested for compliance with security principles [1]. Another basic idea is the use of patterns to guide security at each stage [10]. Patterns are applied in the different architectural levels of the system to realize security mechanisms. This project proposes guidelines for incorporating security from the requirements stage through analysis, design, implementation, testing, and deployment.

A web service development methodology should promote a systematic approach to application development using web services, rather than defining a new software development methodology with the expectation that software practitioners will have to forget their own familiar and established methodologies to re-learn another. A better alternative is to leverage on what is already available and customize that particular methodology to incorporate the specifics of web services.

In general a software development methodology should be able to accommodate refinement throughout the development life cycle in an iterative and incremental approach. The methodology should consist of phases that span from the conception of the need for the web service, to the construction of the web service and finally its deployment. Most methodologies considers security.The only approach that is tailored specifically for web services security is:

PWSSec (Process for Web Services Security) is a web service development methodology that enables the integration of a set of specific stages into the traditional phases of web services-based systems development providing them with security [11]. This process can be applied in the development of new web services-based systems or in systems that are already built. PWSSec is composed of three stages: WSSecReq (Web Services Security Requirements), WSSecArch (Web Services Security Architecture) and WSSecTech (Web Services Security Technologies).

4. THE CURRENT STATUS OF WEB SERVICES STANDARDS AND PRODUCTS

We classified these web services standards into eight groups: XML, Messaging, Description and Discovery, Security, Reliable Messaging, Business Process, Transaction, and Management Specifications. Each group identifies several standards that have similar objectives. There are some standards that are composed by many parts such as XML Schema that has three parts: primer, structure, and datatypes. Usually primer contains basic information to have a better understanding of the standard. The second part may be the framework or core that includes the main structure of the standard, and the other parts may be extended features. There are other standards that depend on others such as WS-Security that uses XML Encryption and XML Digital Signature. Even some other standards may overlap or conflict to each other such as ebXML and UDDI standards that define similar functionalities.

The groups are:

- **XML**. Standards that refer to the syntax and use of this language.
- **Messaging**. This group includes specifications that enable entities to exchange XML messages in a distributed environment.
- **Description and Discovery**. Describe web Services in terms of location, operation, interfaces, and policies, and publish this information in order to be publicly accessed.
- **Security**. Describe how to secure communication between applications through integrity, confidentiality, authentication, and authorization.
- **Reliable Messaging**. Guarantee the delivery of messages even when the the system or network fails.
- **Business Process**. The highest level specifications that specify business process and participants involve in a transaction.
- **Transaction Specifications.** Provide coordination mechanisms when interoperability is needed between different domains.
- **Management Specifications**. Describe how to manage and access web services or other resources located remotely on their networks.

There are a large number of products and tools available on the market that are designed with the objective of giving greater levels of functionality while conforming to industry standards and specification. We evaluated some of the products and development tools available to create web services including their capabilities; with a view of identifying areas that either have no support or can be better enhanced.

5. VALIDATION, CERTIFICATION, AND GOVERNANCE OF WEB SERVICES SECURITY AND RELIABILITY

Systems that support critical systems must be certified to show that they comply with some requirements. This certification must be part of the development process. Certification is based on testing and formal proofs (model checking) for parts of the system.

A development method can be evaluated against the stated goal of covering all stages of the development life cycle using a matrix of concerns vs. development stages [3]. For each concern, the method must include appropriate strategies to address that concern in each stage. Similarly, for each stage, there must be an appropriate response to each concern. The coverage analysis can also evaluate the method's response to the specific issues of legacy, COTS, and outsourced components at the points where they occur (or first appear) in the overall development life cycle. The methodology also needs to be applied to some complex systems and it should show that they can be made more secure according to some qualitative metric (there are no quantitative measures of security).

Patterns are normally evaluated by submitting them to some pattern conference, e.g. Pattern Languages of Programs (PLoP) or EuroPLoP. In these conferences, a pattern paper is developed with the help of a shepherd and then discussed in a workshop by about ten people. The pattern is then published and exposed for criticism.

An important question is the level of security reached by this approach. We cannot prove general security properties of a complete system produced this way. However, we have applied a systematic approach and validated patterns and we expect to eliminate in this way a large variety of threats. A qualitative analysis can be performed to show that we have stopped or mitigated all the identified threats. There will still be threats based on the actual code but those can be handled by other means.

The distributed and possibly heterogeneous nature of SOA makes governance a complex task. SOA appliances can be used to collect metadata and provide analysis and management. SOA requires cooperation and coordination between business and information technology (IT) as well as among IT departments and teams for smooth functioning. This cooperation and coordination is provided by SOA governance which lays out the rules and policies around service creation, service discovery, service identification and reuse. It defines the service-level agreements (SLAs) for how services should perform, so that both the consumers and the providers know their limitations and expectations. In summary, governance gives the providers and the consumers the same view of the service's quality. Since SOA extends interactions beyond the enterprise boundary, the governance of SOA must interact with similar groups in other organizations to achieve a common set of standards for communication across the enterprise boundary [12].

Governance means establishing and enforcing how a group agrees to work together in co-ordination. Specifically, governance is the establishment of:

- Chains of responsibility to assign duties and rights to people
- Measurement to gauge effectiveness
- Policies to guide the organization to meet its goals
- Control mechanisms to ensure compliance
- Communication to keep all required parties informed

In other words, we can say that governance determines what decisions are to be made, who should be making the decisions and what policies should be considered before making any decisions. In practice, SOA governance guides the development of reusable services and establishes how the services should be designed and developed and how those services should change over time. It establishes agreements between the providers of services and the consumers of those services, telling the consumers what they can expect and the providers what they're obligated to provide. SOA governance is a set of practices that is applicable throughout the various stages of a service-oriented architecture such as service definition, service deployment lifecycle, service versioning, service migration, service registries, service message model, service monitoring, service ownership, service testing and service security. SOA governance ensures that all of the independent efforts whether in the design, development, deployment or operations of a service should come together to meet the enterprise SOA requirements.

SOA governance is enacted by an SOA center of excellence (COE) which is a board of knowledgeable SOA practitioners who establish and supervise policies to help ensure an enterprise's success with SOA. The COE establishes policies for identification and development of services, establishment of SLAs, management of registries, and other efforts that provide effective governance. Governance policies can be enforced through a combination of an enterprise service bus (ESB) and a service registry. A service can be exposed so that only certain ESBs can invoke it. Then the ESB/registry combination can control the consumers' access, monitor usage, measure SLA compliance, and so on. This way, the services focus on providing the business functionality and the ESB/registry focuses on aspects of governance.

6. CONCLUSIONS

Web services security has been its weak point and the cause of its rather slow adoption. Work is needed to make this security predictable and systematic during the development of new applications. The methodologies and patterns discussed in this work go in that direction and we expect they will make a significant contribution but we need more patterns and testing the methodology in real environments. We identified some promissory directions, involving formal methods and MDA and we will pursue these directions.

Many of the web service products and tools discussed offer several of the same features; also many conform to a few of the most common web services standards such as SOAP, XQUERY and XML. However, the problem being faced is how to select the right web services product or tool which best suit a designer’s needs. This is very hard to determine at present since many companies do not explicitly state the features and standards which are supported by their products or tools, more over it is very time consuming to acquire this information. Additionally many products may only conform to a few standards; for example from this survey it is observed that many products are not compliant with WS-Reliability standard among others. A possible solution in overcoming this problem is using web service patterns in the implementation and design of web services products and tools.

Web services can be used in critical applications but careful adherence to standards is necessary to assure a strong application. Applications developed in non-systematic ad hoc ways will not be able to provide the required levels of security and reliability. In particular, web services can be extremely valuable in military applications because these involve heavily distributed systems with security and reliability being basic requirements.

We produced several security patterns, including [4, 8, 13, 14]. Two other papers and two MS theses also resulted from this work. Our complete report to DISA includes tables of standards and products as well as sections on architecture, patterns, and certification [15].

ACKNOWLEDGEMENTS

This research was partially funded by the U.S. Department of Defense.

REFERENCES

[1] E.B. Fernandez, M.M. Larrondo-Petrie, T. Sorgente, and M. VanHilst, "A methodology to develop secure systems using patterns", Chapter 5 in *Integrating security and software engineering: Advances and future vision*, H. Mouratidis and P. Giorgini (Eds.), IDEA Press, 2006, 107-126.

[2] E.B. Fernandez, J.C. Pelaez, and M.M. Larrondo-Petrie, "Attack patterns: A new forensic and design tool", Procs. of the Third Annual IFIP WG 11.9 Int. Conf. on Digital Forensics, Orlando, FL, Jan. 29-31, 2007. www.cis.utulsa.edu/ifip119

[3] M. VanHilst, E.B.Fernandez, and F. Braz, "A multidimensional classification for users of security patterns", accepted for the *Journal of Research and Practice in Information Technology.*

[4] I. Buckley, E.B Fernandez, “A Survey of Fault Tolerance Patterns”, Department of Computer Science and Engineering, Florida Atlantic University, 2007

[5] OASIS, Web Services Business Process Execution Language Version 2.0, 11 April 2007,

http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[6] W3C, Web Services Description Language (WSDL) 1.1, 15 March 2001, http://www.w3.org/TR/wsdl

[7] W3C, Web Services Description Language (WSDL) Version 2.0 SOAP 1.1 Binding, 26 June 2007, http://www.w3.org/TR/wsdl20-soap11-binding/

[8] I. Buckley and E.B.Fernandez, “Web Services reliability patterns”, Department of Computer Science and Engineering, Florida Atlantic University, 2007, sent for publication.

[9] OASIS, eXtensible Access Control Markup Language (XACML) Version 2.0, 1 Feb 2005,

http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[10] M. Schumacher, E.B.Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, Security Patterns: Integrating security and systems engineering, Wiley 2006.

[11] C. Gutierrez, E. Fernandez-Medina, and M. Piattini, “PWSSec: Process for Web Services Security“ , IEEE International Conference on Web Services (ICWS’06), 2006, 213-222.

[12] T. Mitra, “A case for SOA governance,” IBM DeveloperWorks Technical Library, 15 August 2005.

http://www.ibm.com/developerworks/webservices/library/ws-soa-govern/

[13] K. Hashizume, E.B.Fernandez, and S. Huang, “The WS-Security pattern”, sent for publication.

[14] K. Hashizume and E.B.Fernandez, “The XML Signature pattern”, sent for publication.

[15] E. B. Fernandez, M.M. Larrondo-Petrie, M. VanHilst, I. Buckley, K. Hashizume, and M.Garg, ”Web services security and reliability: Standards and industrial practice”, Department of Computer Science and Engineering, Florida Atlantic University, January 2009.

Autorización y Renuncia

Authorization and Disclaimer