Web Services-based Security Requirement Elicitation

Carlos Gutiérrez [†], Eduardo Fernández-Medina [††], Mario Piattini [††]

Summary
Web services (WS, hereafter) paradigm has attained such a relevance in both the academic and the industry world that the vision of the Internet has evolved from being considered as a mere repository of data to become the underlying infrastructure on which organizations' strategic business operations are being deployed [1]. Security is a key aspect if WS are to be generally accepted and adopted. In fact, over the past years, the most important consortiums of the Internet, like IETF, W3C or OASIS, have produced a huge number of WS-based security standards.
Despite this spectacular growth, a development process that facilitates the systematic integration of security into all subprocesses of WS-based software development life-cycle does not exist. Eventually, this process should guide WS-based software developers in the specification of WS-based security requirements, the design of WS-based security architectures, and the deployment of the most suitable WS security standards. In this article, we will briefly present a process of this type, named PWSSec (Process for Web Services Security), and the artifacts used during the elicitation activity, which belongs to the subprocess WSSecReq aimed at producing a WS-based security requirement specification.

Key words:
Software Engineering, Design Methodology, Software Process, Application Information Security

1. Introduction

Security is a main concern when developing systems whose operational infrastructure is based on public networks such as the Internet.

WS-based systems are founded on Internet protocols so security should be one of the main issues to be addressed when designing applications based on this paradigm.

A huge number of WS-based security standards have been developed by a numerous set of diverse consortiums. A great effort and a solid background in computational security theory are necessary in order to obtain an in-depth knowledge of all of them. In addition, knowing what specific set of WS standards should be used in a certain WS-based system requires a previous knowledge of the security requirements that the security mechanisms specified in those standards will address.

As a consequence, one of the major problems that developers have to deal with is to come up with a complete specification of the WS-based security requirements of their WS-based systems.

In order to solve this problem, we have defined the PWSSec (Process for Web Services Security) process [1]. This process is made up of 3 subprocesses. The first subprocess, named WSSecReq (Web Services Security Requirements) is aimed at producing the aforementioned WS-based security requirements specification. In particular, its first activity of elicitation uses a set of reusable artifacts that guides developers in the task of identifying security requirements from the piece of functionality whose security is to be analyzed.

This article's main purpose is to describe this set of security artifacts showing how they can be used in an aligned, reasoned and coordinated approach to semi-systematically specify the security requirements of a WS-based system.

The rest of the article is organized as follows: in section 2, an overview of PWSSec process will be presented; in section 3, a complete description of the mentioned artifacts will be provided; section 4 will show how traceability is assured; in section 5, related work to that described in this paper is discussed; and, finally, in the last section, conclusions as well as future research will be mentioned.

2. PWSSec – Process for WS Security

PWSSec [1] has been created to facilitate and orientate the development of WS-based security systems so that a complementary subprocess comprising security [2] could be easily integrated into each one of the traditional subprocesses for the construction of this kind of systems [3].

Figure 1 illustrates the set of subprocesses PWSSec is made up. Each one of the subprocesses describes its inputs, outputs, activities, actors and, in some cases, guides, tools and techniques thereby complementing, improving and facilitating their practical application.

The WSSecReq subprocess main purpose is to produce, by means of a systematic approach, a specification (or a part of it) of the security requirements of the WS-based system. A deeper explanation of this subprocess has been presented in [1].

The WSSecArch subprocess is aimed at allocating into

[†] The author is with Correos Telecom, C\Conde de Peñalver, 19bis – 28006 Madrid, Spain.
[††] The authors are with Alarcos Research Group. Information Systems and Technologies Department UCLM-Soluziona Research and Development Institute. University of Castilla-La Mancha, Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain.

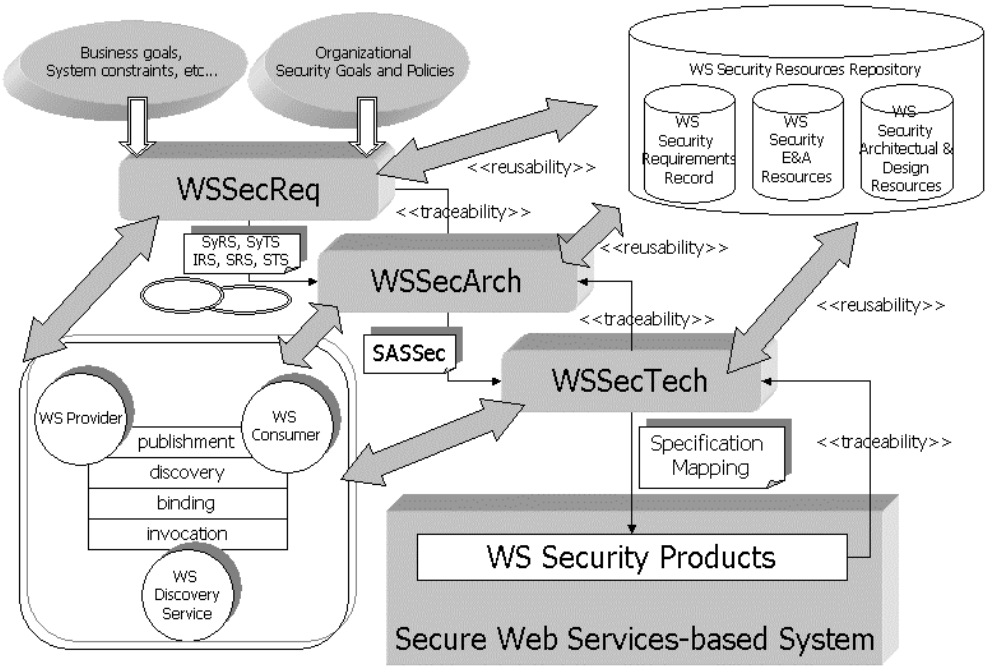

Fig 1. Subprocesses and main security artifacts of the PWSSec process.

a WS-based security architecture the security requirements specified in the previous section. This security architecture will be equipped with the necessary security architectural mechanisms to achieve the identified security requirements. A case study that demonstrates how this WS-based security architecture can be designed has been presented in [4].

The WSSecTech subprocess' main objective is to identify the set of WS-based security standards that will implement the architectural security mechanisms identified in the previous subprocess.

3. WSSecReq – Security Requirements for WS

In this section, we will explain all the security artifacts which the elicitation activity of the WSSecReq subprocess is based on.

Sometimes, we will present concrete examples where these artifacts are applied in practice. The examples of concrete artifacts shown here are based on the classical use case 'Place Order'. In this use case, a WS-based system of a retailer organization (primary actor) and a WS-based system of its supplier organization (secondary actor) participate [3]. This use case consists of one request/reply message interaction between the WS-based systems of both organizations. When the WS-based retailer system detects that any of its products is out-of-stock, it sends a request (and it gets blocked until a response is received) of stock replenishment to the WS-based system of the supplier organization.

3.1 WSSecReq overview

In this section, we will describe the WSSecReq subprocess (see Figure 2), of PWSSec process.

Objectives and general considerations

The main purpose of this subprocess is to produce a specification (or a part of it) of the security requirements of the target WS-based system. Its input is composed by a specification of the scope that we want to accomplish during the current iteration (e.g.: if we have a Use Case Model available, we can select those that we want to cover and use them as an input for the iteration), the business and security goals defined for the system as well as the part of the organizational security policy that we estimate that may impact on the system design.

PWSSec Process
Sub-process P1 – **WSSecReq**
Activity A1.1: Elicitation
Task T1.1.1: Decide granularity level and identify the fragment of functional software whose security will be analyzed.
Task T1.1.2: Identify the IBM WS-based business pattern.
Task T1.1.3: Identify the IBM WS-based application pattern.
Task T1.1.4: Identify possible threats at the business-level.
Task T1.1.5: Identify possible threats at the application-level.
Task T1.1.6: Relate business and application-level threats.
Task T1.1.7: Identify and evaluate threats.
Task T1.1.8: Identify the type of attackers and their possible types of attack.
Task T1.1.9: Assess the impact of the attacks.
Task T1.1.10: Estimate and prioritize the security risks.
Task T1.1.11: Determine the behaviour the system should have for each attack.
Task T1.1.12: Specify security requirements.
Activity A1.2: Analysis
Task T1.2.1: Identify conflicts due to composition or integration scenarios.
Task T1.2.2: Remove redundant and refine ambiguous requirements.
Task T1.2.3: Classify elicited security requirements.
Task T1.2.4: Identify inclusion/exclusion relationships among the requirements.
Task T1.2.5: Update the E&A Repository.
Activity A1.3: Specification
Task T1.3.1: Instantiate the templates of the security requirement specification documentation.
Task T1.3.2: Arrange set of security requirement specifications adhering to SIREN approach.
Activity A1.4: Verification and Validation
Task T1.4.1: Internal Verification.
Task T1.4.2: External Validation.

Fig 2. Activities and tasks of WSSecReq subprocess.

Principles

Traceability is addressed by means of a coordinated and reasoned use of a set of security artifacts. These artifacts and their application have been detailed in [5]. In order to support the reusability principle, this subprocess is supported by two repositories:

- *WS Security E&A Resources*, that contains all the aforementioned security artifacts (special mention deserves the SIREN-based Security Web Services catalogue where a set of WS security requirements templates are being gathered [6]). These artifacts can be reused across different WS-based systems.

- *WS Security Requirements Record* that contains a set of generic security requirements that can be applied to WS-based systems within diverse domains and that IBM Business/Application Patterns elements are related to [7].

Both repositories are constantly being brought up-to-date.

Input

The input to this subprocess are:

- A specification of the piece of software functionality whose security will be analyzed. WSSecReq treats security analysis as a micro-process which is performed at each level of abstraction and for each increment [8].

- The business and security goals, constraints and assumptions defined for the system, as well as the part of the organizational security policy that may impact on the system design.

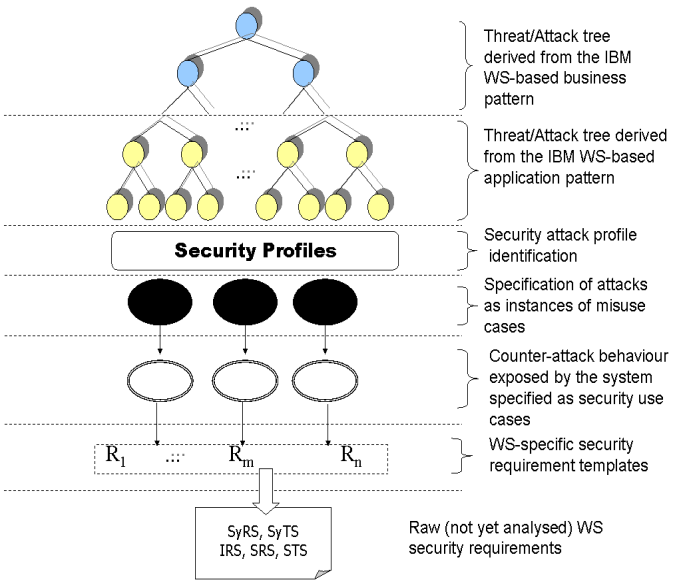

Fig 3. Coordination of products in the activity of elicitation of the WSSecReq subprocess.

Output

The output is basically formed by:

- A threat attack tree containing threats, and their possible attacks, at both business and application-level [9] and associated with the IBM's WS business and application pattern [3] identified from the analyzed fragment of functionality.
- Every built attack tree's leaf will show a threat [10] that can be refined by a set of attack scenarios, defined as misuse cases according to [11, 12], organized into attack profiles [13], and represented according to the Quality of Service UML profile [14].
- In turn, every misuse case will hold a related a set of security use cases, according to Donald G. Firesmith [15], that state how the system should respond to the associated misuse case. A formal specification of the security requirements for the scope of the system based on SIREN [6, 7]. These requirements will have been derived after instantiating the WS security requirements templates associated with every security use case. A practical application of all these artifacts

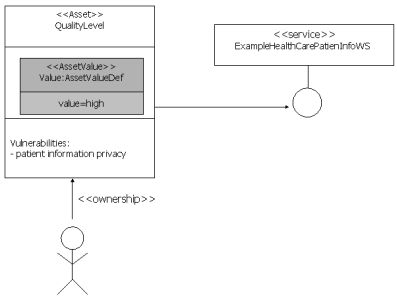

Fig 4. Level of Quality modelled as an asset related to a WS by means of the QoS profile.

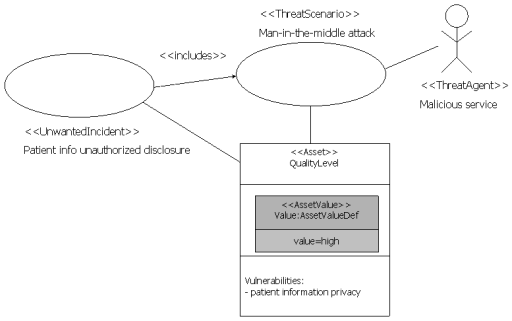

Fig 5. The asset QualityLevel is related to the potential identified attack, which in turn is related to its attacker. In addition, the asset QualityLevel shows a relationship with the unwanted situation that may arise when its potential attack is successful.

was presented in [5].

Actors

- Domain Expert Group, composed of people from the different business entities being "networked" [16]. These people will be in charge of providing the organizational security policy to be taken into account and coming up with a unified organization security policy of the system to be built up.
- Requirement Engineering Team, responsible for leading the completion of the activities, and their tasks. This actor should incorporate specialists in security requirement engineering whose main purpose will be to instantiate all the security artifacts handled during this activity (i.e.: threat attack trees, misuse and security use cases, etc.).
- Security Team, contributing with its knowledge about security, and the specific security infrastructures of the involved enterprises, during the development of the sub-process.

Activities

A1.1.Elicitation. The elicitation activity is supported by a detailed study of security for each WS business service identified and considered in the current iteration. This activity is inspired in the risk analysis and management process known as Operationally Critical Attack, Asset, and Vulnerability Evaluation [SM] (OCTAVE) [17].

This activity's tasks (see Figure 2) use a set of security artifacts in a gradually form as can be seen in Figure 3. Following, a brief explanation of the security artifacts used during elicitation's activity progress is specified.

In Task **T1.1.1,** a specification of the scope and abstraction-level of the piece of software functionality whose security will be analyzed is determined. WSSecReq observes security analysis as a micro-process which is performed at each level of abstraction and for each increment [8]. As we are dealing with WS-specific security requirements, the core artifacts will mainly belong to the system and to the application architecture level of abstraction, i.e. WS and their interactions. However, as we will see later on, WSSecReq may also be used to analyze security at higher levels of abstraction, for instance, the business level one. For example, WSSeqReq subprocess's input could be specified at a low level of abstraction (application architecture) when it is composed of a (small) set of WS and operations which are within the scope of the current iteration (in this case the "security enhanced core artifacts" will be the WS and their interactions). On the other hand, it could be specified as a set of high-level functional requirements that describe how a group of responsibilities should be addressed by the WS-based system.

Abstract TA Tree related to the WS-based business pattern Extended Enterprise
ID: A3Business-EE
Objective:
1. Cause harm in the execution of service x
OR/AND 1. Attack Business Entities
…
2. Attack Network
…
3. Attack Business Rules
…
4. Attack Business Process Interactions
…

Fig 6. Abstract TA tree associated with the WS-based business pattern named Extended Enterprise.

Place Order Business Process TA Tree
ID: A3Business-EE-1
Objective:
1. Cause harm during Place Order Business Process Execution.
OR/AND 1. Attack Business Entities.
OR/AND 1. Attack Retailer Organization.
OR/AND 2. Attack Supplier Organization.
2. Attack Network.
OR/AND 1. Attack Retailer Organization's Network Infrastructure.
OR/AND 1. Attack Retailer Organization's Internet Provider.
…
OR/AND 2. Attack Supplier Organization's Network Infrastructure.
OR/AND 1. Attack Supplier Organization's Internet Provider.
…
3. Attack Business Rules.
…
4. Attack Business Process Interactions
…

Fig 7. Concrete TA tree associated with the WS-based

In task **T.1.1.2** and **T1.1.3** no security-specific artifacts are used.

In tasks **T1.1.4**, **T1.1.5**, **T1.1.6** and **T1.1.7**, potential threats at both, business and application-level are identified and organized as threat/attack trees and modelled with the Quality of Service UML profile [14]. Threats will take place in attack scenarios over WS (the interface with the service itself as well as the set of actions with internal elements such as databases or directories that

the interface executes to complete that service) according to the type of SOA abstract interactions that the service must perform (publishing, discover, binding and invocation). Normally, there will be two groups to be analyzed for each WS: the first one formed by binding and publishing processes of the WS and the second one formed by discovery, binding and invocation processes of the WS. Apart from defining the attack scenarios (as misuse cases), we define its associated security scenarios (as security use cases) that specify how the WS should respond in order to prevent (or at best to mitigate) the attack. In Figure 4, QoS UML profile has been used to model the level of quality (depicted with stereotype <<Asset>>) associated with certain business WS called ExampleHealthCarePatienInfoWS. In Figure 5, QoS UML profile modelling of unwanted incidents (exploited vulnerabilities by means of successful attacks) is shown.

In task **T1.1.8**, we use every IBM WS-based application pattern identified in the current iteration as an index to the Attack Profiles [13] contained in the *WS Security E&A Resources* repository. Thus, we'll obtain the set of Attack Profiles to be used when discovering the types of attacks and attackers. An Attack Profile contain, among other things (security artifact Attack Profile will be explained in section 3.2) a set of potential types of attackers and attacks expressed as misuse cases' templates [11, 12]. Every misuse case is related to one or more threats (that were derived from the same IBM WS-based application pattern the Attack Profile containing the misuse case is related to). Therefore, threats in the threat/attack tree developed in **T1.1.4-T1.1.7** will be refined as different types of potential attacks.

In tasks **T1.1.9** and **T1.1.10,** for each vulnerable WS, we must determine the negative impacts that could appear if the attacks against this WS would happen and how the impact of this attack could be spread to the services interacting with it as well as to the underlying infrastructure (e.g.: ERP systems, databases, directory services, etc.). In addition, security risk analysis is performed. Security risk is the potential risk of causing damage to an estimated WS from the addition (taking into account all the relevant threats) of the negative impact of the caused damage multiplied by the probability of this impact to happen [18]. In this way, we can detect which WS are more relevant in terms of security and we can dedicate more resources to them during current and future

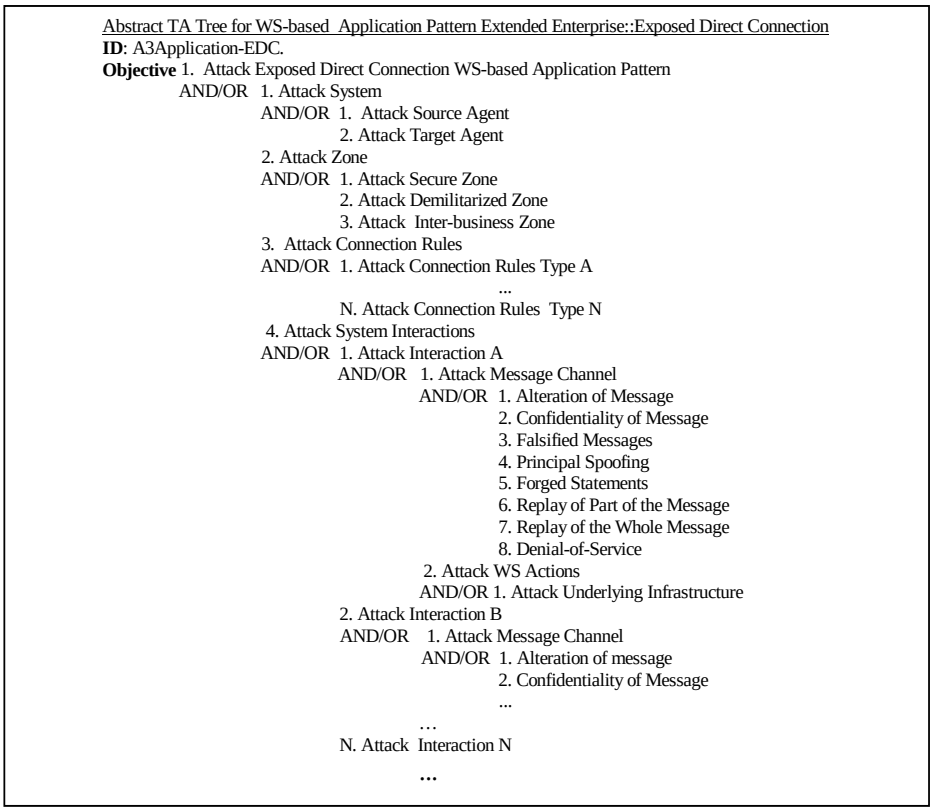

Fig 8. Abstract TA tree associated with the WS-based application pattern named Exposed Direct Connection.

iterations.

In **Task T1.1.11,** we specify the expected behaviour of the system when attacks specified as misuse cases are manifested. Every misuse case template in the *WS Security E&A Resources* holds a relationship with one or more security use cases' templates. For each misuse case considered in **T1.1.8**, its associated security use case will be instantiated. In turn, every security use case holds a relationship with one or more security requirement templates. This association determines the security subfactors [19] to that should be contemplated.

Finally, in **Task 1.1.12**, security requirement specification is done. This task accomplishes the next steps:

- Using instantiated security use cases as input, we'll retrieve from the repository of reusable requirement templates, the set of templates to be used for each security use case, security risk and associated subfactor. In our case, an example of template for information privacy could be as follows: "*The [WS consumer, WS provider, WS discovery] will guarantee the non revelation of [type | identifier] of information without the express consent of its owner to [WS consumer, WS provider, WS discovery] during the execution of [set of interaction/use cases] according to the criterion and measures specified in the table [table]"*.

- Determine the security criterion in order to introduce its parameters into the template. The security criterion specific for the WS which the template will be used for, determines how the degree of presence of a certain security subfactor will be measured. The security criteria can be determined according to the services, types of attackers or identified attacks and security risks. In the template of our example, we have determined that the criterion will be: "Minimum number of attributes kept private".

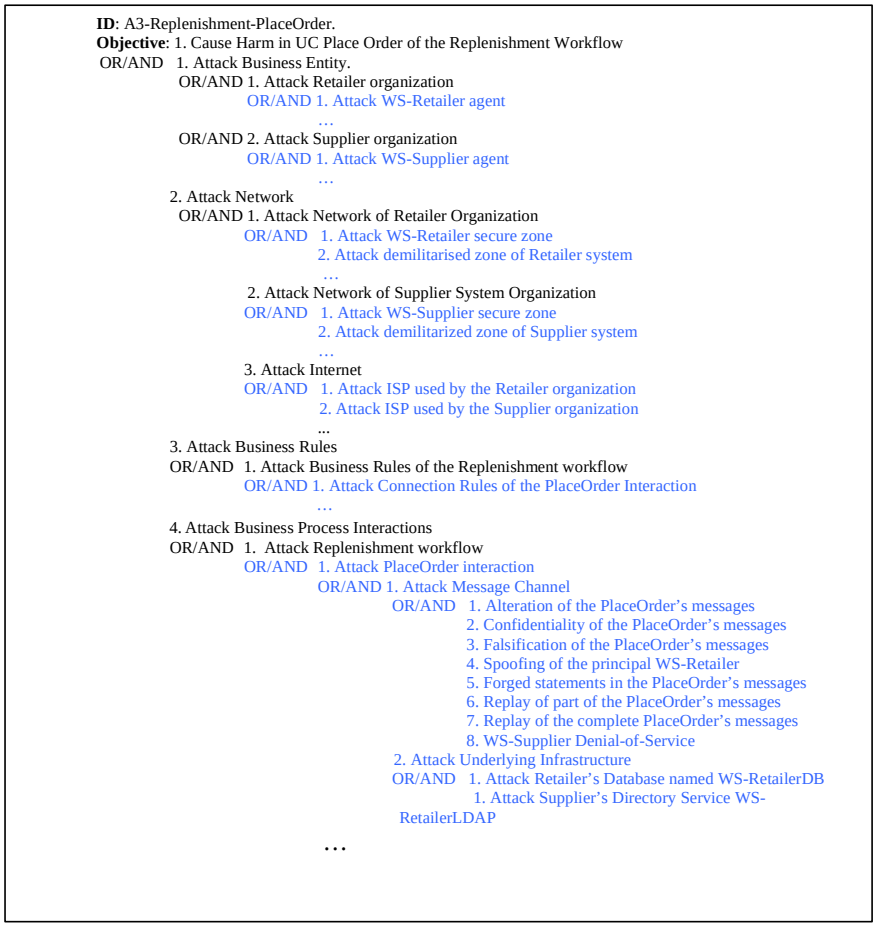

Fig 9. Concrete TA tree resulted after combining the business-level TA tree and the application-level TA tree.

- Determine the suitable security metrics that measures the existence of the chosen security criteria and to introduce the quality metrics into the template. In our case, metrics is a percentage.
- According to the security risk identified for a certain service, to determine the minimum acceptable level of metrics for the chosen criterion, that limits to an acceptable level the associated risk and to introduce the template required level. In our example, the accepted level is 99.99 %.
- Specify the security requirement by instancing the template from the selected parameters in the last three steps. We'll show an example of this instantiation in section 3.2.

A1.2.Analysis. The analysis activity basically consists of identifying the possible conflicts that could arise among the security requirements. Following, tasks defined within this activity are described.

In **T1.2.1** we must identify potential conflicts that could arise from two possible perspectives: i) security requirements conflicts within composition scenarios [20]: If there are new services built by composition, we must verify that these new services do not violate any of the identified security requirements; ii) security requirements conflicts within integration scenarios [21, 22]: i) external services, governed by third parties, which we want to integrate with; ii) inherited systems that we want to offer an interface based on a WS for. In PWSSec no specific (formal) method is mandated to accomplish this task. The approach that has been used when applying PWSSec to real case studies (see [4, 40, 41]) are based on peer reviews with this activity's actors.

In **T1.2.2**, elimination and refinement of redundant and ambiguous security requirements is performed respectively.

In **T1.2.3** a security requirements classification is done. The analysis classifies security requirements in terms of:

- Business/application requirement type.
- System security, software or interface requirement type.
- Security subfactor addressed by the security requirement.
- In case of being an application-level requirement type, whether it is a security requirement on the publishing, discovery, binding or invocation process.

In **T1.2.4**, traceability relationships between the different security requirements are identified [7].

Finally, in **T1.2.5** we should update the WS Security E&A Resources with possible new or modified security artifacts developed during the execution of the activity.

A1.3.Specification. This activity basically consists of documenting the WS security requirements. Requirements specification, based on IEEE std. 1233, 12207.1, 830 standards, and is supported by the idea of the use of a set of requirements and a hierarchical structure of reusable documents defined in SIREN [7]. Tasks **1.3.1** and **1.3.2** (see Figure 2) are performed by the Requirement Engineering Team and the Security Team. The template of the documentation to be elaborated is extracted from the repository *WS Security E&A Resources*. As part of this activity, the set of documents to be created and their hierarchical arrangement (as determines SIREN approach) must be outlined.

A1.4. Validation & Verification. Basically, this activity's main are perform internal validation and external verification as expressed by [23]. Internal verification must identify potential conflicts among security requirements and the rest of requirements, and detect incomplete, inconsistent, incomprehensible, or ambiguous requirement specifications. Además debemos comprobar que los requisitos de seguridad distribuidos por los diferentes documentos de especificaciones no son redundantes, son consistentes y se encuentran bien clasificados. Several techniques can be applied when performing this task such as peer reviews, checklists [24], tools [25] or Fagan's method [26].

4. Traceability in WSSecReq

In WSSecReq, traceability characteristic is mainly present in the elicitation activity. This activity specifies a set of tasks that will produce a set of security requirements related to the piece of software functionality under analysis (SuA). The set of activities and their tasks are independent of the underlying technologies, being the security artifacts used in them which tie the activities and tasks to the WS-based paradigm.

In this section, we will explain the security artifacts involved in this activity and how they are linked together to obtain full traceability between the WS, whose security is under analysis, and their elicited security requirements.

4.1. WS-based business and application patterns

In [3], a catalog of WS-based business, integration, application, composite and runtime patterns are presented. This catalog of WS-based patterns offers us a complete pattern-based design solution space for modelling WS-based systems. In our work, we use these patterns as a reference and starting point for identifying the set of

Table 1. Attack profiles associated with the WS-based application pattern Exposed Direct Connection.

Business Pattern	Application Pattern	Element	Variation	Attack Profile	ID
Extended Enterprise	Exposed Direct Connection	Interaction	Message-based Variation	*WS Message-based Interaction with no Acknowledgement*	AAP-1-1
			Invocation-based Variation	*WS Message-based Interaction with Acknowledgement*	AAP-1-2

potential threats that should be assessed during the elicitation activity. Basically, these patterns define a set of elements, and their interactions. Thus, threats on these elements and interactions are studied and considered from the very beginning.

First of all, the WS-based *business* patterns underlying the design of the functionality whose security is under analysis are identified and instantiated for the specific system. Notice that if we have already made an analysis at the business level, we may not need to identify and instantiate this type of WS-based pattern. In this section, we will assume that we have already identified a WS-based business pattern.

For every WS-based business pattern, a WS-based application pattern can be selected. If we have not specified a WS-based business pattern, we may identify the WS-based application pattern straight forward from the functional software architecture. Then, the WS-based application pattern which the SuA is based on is selected and put into the context of the system. The identification of the WS-based application pattern assumes that there exists a functional architecture where, at least, a set of core WS and interactions have already been defined. For each WS-based business and WS-based application pattern, we have defined an Abstract Threat/Attack (TA) tree [9, 13] that shows how the elements and their interactions - as defined by the WS-based patterns - are threatened.

Our concept of threat and attack is based on the Internet Glossary (RFC 2828) [27].

4.2. Abstract and concrete TA trees

We have adapted the security attack trees, as defined in [9, 13], to the context of security WS-based systems. For every WS-based business and application pattern, we have established a relationship with an Abstract TA tree. Thus, once both WS-based patterns have been identified and instantiated for the part of the SuA of the current iteration, a tree-like structured set of threats at both, the business and the application level, is semi-systematically obtained. Firstly, the abstract TA tree associated with the WS-based business pattern will be instantiated.

In consequence, a concrete business-level TA that is specific for the current iteration's SuA is defined. In Figure 6, an abbreviated example of the abstract TA tree associated with the IBM's WS business pattern named Extended Enterprise is shown. The set of threats structured in this tree have been extracted from those defined in the methodology for the risk analysis and management adopted by the Spanish Public Administration. This

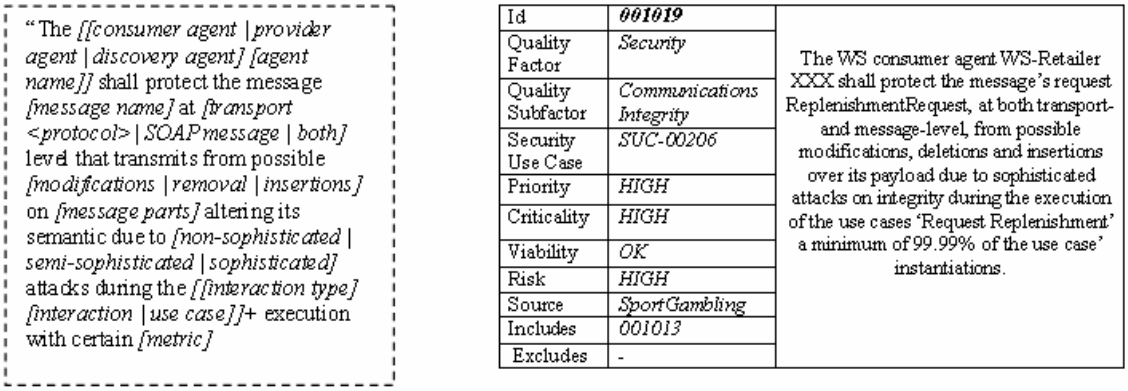

Figure 10. Integrity security requirement template and an example of its instantiation.

methodology's name is Magerit2 [28] and is compliant with the Common Criteria Framework. In Figure 7, an example of instantiation of the mentioned abstract TA is depicted. The same process will be performed for the WS-based application pattern so that a concrete application-level TA tree can be produced. In Figure 8, the abstract TA tree associated with the WS-based application pattern named *Extended Enterprise::Exposed Direct Connection* is shown.

Branch 1.1, known as *Attack System*, will refine branch 1.1 *Attack Business Entities* of the A3Business-EE, 1.2 will refine branch 1.2 of the A3Business-EE, and so forth. Likewise, branch 1.4 *Attack Business Process Interactions* of the A3Business-EE will refine branch 1.4 Attack *System Interactions* The set of threats which appear under branches 1.4.x.1 have been taken from [10]. Once both concrete TA trees have been developed, they will be combined to obtain a single TA tree that groups the set of threats to be considered within the selected fragment of functionality.

An example of the resulting TA tree once both TA trees, business and application level trees, have been combined is shown in Figure 9. This combination, as the result of applying a set of rules associated with both TA trees establishes a derivation relationship among the threats of the business TA tree and threats of the application TA tree.

We should highlight the fact that, thanks to this adaptation of the attack trees from [9, 13] and the relationship established with the WS-based business and application patterns, we are making it possible not only to consider the aspects of security of the interactions of the WS security agents themselves but also to take into account possible attacks on the provider and consumer organizations, on the network services (e.g.: attacks on the Internet Service Providers of any of the participating business entities) or the infrastructure in use, along with other elements at the organizational and business level.

4.3. Attack Identification

The next step will consist of refining the leaf-nodes of the TA tree, i.e. further specification of the threats by means of concrete attacks. The threats themselves are of no significance if there are not attacks which may bring them to fruition. It is the right time, then, to identify the set of possible attacks which could occur, for each of the threats identified. To do this, use will be made of the concept of Attack Profile described in [13]. The attack patterns set out in this work seem not particularly formal, as compared to the misuse cases in [29]. As both artifacts have the same purpose, i.e. to define the sequence of steps of successful attacks on the system, we have opted to employ the latter when defining the attack profiles. Basically, an attack profile contains a set of abstract misuse cases that apply to a reference model defined within the profile. Thus, interactions in every WS-based application pattern have one attack profile related. Every WS-based application pattern has one or more attack profiles related to it which state the potential attacks that could be targeted at them. For instance, for the WS-based application pattern *Exposed Direct Connection*, the set of attack profiles exposed in Table 2 has been defined. Every attack profile gathers a set of abstract misuse cases that focuses on a particular element defined within the reference model specified for the questioned profile. In Table 1, both attack profiles are interaction-centered, i.e. the attacks they contain are focused on exploiting any vulnerability that may be deduced from the analysis of the messages exchanged within the interaction and from the nature of the interaction itself (e.g.: synchronous vs. asynchronous, message exchange pattern in use, etc.). Other attack profiles, which are connection rules-centered or zone-centered, have been specified. In our example, the PlaceOrder interaction follows a request-reply message exchange pattern. An uncontrolled network, i.e. the Internet, is the context that should be assumed for it. These are the variants specified for this profile:

- WS Provider and WS Consumer Organizations. In this study, these are the Supplier and the Retailer Organization, respectively.
- WS Provider Agent and WS Consumer Agent. In our case, these are the WS-Supplier and WS-Retailer agents, respectively.
- The name of the operation to be performed, here known as PlaceOrder operation, which is WSDL (Web Services Description Language)-classified [30] as a request/reply message exchange pattern. A set of misuse cases has been defined as a result of analyzing the threats enumerated in branch 1.4.x.1.y (interaction attacks) of the A3Application-EDC. Misuse cases specify the possible attack scenarios that materialize the threats which they are associated with. We have abstract misuse cases and concrete misuse cases. As mentioned above, the former are grouped into attack profiles, while the latter are instances of the former and set out the sequence of steps for a given attack. Two abstract misuse cases described in this profile are listed below:

  i) **Misuse Case Attack on SOAP's Message Semantic (AMUC-1-1-1),** which refines the threat represented by branch 1.4.x.1.1 (Alteration of the Message) of the A3Application-EDC in the WS-based application pattern *Exposed Direct Connection* (see Figure 8).

Table 2. Abstract misuse case 'Attack to the Semantic Content of the SOAP Message'.

Name of Abstract Misuse Case: Attack on the Semantic Content of the SOAP [message | interaction] [message | interaction name]		
ID: *AMUC-1-1-1*		
PROBABILITY *[HIGH][MEDIUM][LOW]*		
Summary: the attacker type *[attacker type]* gains access to the *[message | interaction] [name]* exchanged by the *[consumer | provider | discovery]* agent *[agent name]* and the *[consumer | provider | discovery]* agent *[agent name]* and *[modifies | deletes | inserts [part]*]* of the message at the *[transport | SOAP]*-level situated in the *[header | body | attachment]* with the object of *[objective]*.		
Preconditions: 1) The attacker has physical access to the message. 2) The attacker has clear knowledge of the structure and meaning of the message.		
Interactions of the Consumer Agent	**Interactions of the Misuser**	**Interactions of the Provider Agent**
The Consumer Agent *[agent name]* sends the message *[name of message]*		
	The attacker *[type of attacker] [name of attacker]* intercepts it	
	The attacker *[type of attacker] [name of attacker]* identifies the part to modify and *[deletes | replaces | adds]* information	
	The attacker *[type of attacker] [name of attacker]* forwards the message to the Provider Agent *[agent name]*	
		The Provider Agent *[agent name]* receives the message*[name of message]* and processes it erroneously due to the altered semantic content.
Postconditions 1) The system will remain in a state of error with respect to the original intentions of the Consumer Agent *[name of consumer agent]*. 2) In the register of the system in which the Provider Agent *[name of provider agent]* was executed the request received with an altered semantic content will be reflected.		

ii) **Misuse Case Attack on SOAP Message's Authenticity (AMUC-1-1-2)**, which refines branches 1.4.x.1.3, 1.4.x.1.4 and 1.4.x.1.5 of the A3Application-EDC (see Figure 8).

Finally, the possible attackers, primary actors in the stated abstract misuse cases, are (extracted from the attack profile):

- WS Provider Malicious Agent: the WS-Provider agent may not behave as expected and perform illicit activities such as revealing the identity of buyers for its own benefit (selling this information, creating buyer profiles to personalize offers, etc.).

- WS Intermediary Malicious Agent: in the SOAP architecture, which Web-based services systems are based on, the figure of the intermediary SOAP nodes appears. These nodes can process messages while traveling along their path. Malicious and not-expected behavior of the possible intermediaries located along the message's path must be considered.

- External Attacker: this is an attacker who has the ability to perpetrate all the attacks we have pointed out from the Internet. The risk from this type of attacker is extremely high, due to how unpredictable and uncontrollable the Internet is.

In Table 2**,** an abbreviated example of an abstract misuse case is presented. As it can be seen, it is highly parameterized; therefore it is not application-specific and can be reused across different WS-based systems and problem domains.

4.4. Specification of System Security Behaviour

Every abstract misuse case holds a relationship with one or more security use cases [15, 31]. Security use cases define a sequence of steps which allow the system to prevent, detect or react to each of the attacks which take place in the form of an instance of the misuse cases they are associated with.

4.5. Specification of Security Requirements

Each abstract security use case holds an association with one or more templates of WS-based security requirements, which should be instantiated in order to obtain the final security requirements. The security requirement follows the basic guidelines proposed in [19]. In Figure 10, an example of a WS-based security

requirement is shown. This template is associated with the abstract security use case presented in Table 2.

The steps that should be followed when instantiating the WS-specific security template are explained in detail in [1].

5. Related work

At present the biggest effort is being applied in the area of advancing WS security-related standards and specification definition. This effort has caused the existence of a vast number of specifications and standards. Due to this multiplicity and diversity of proposals it has become very difficult for an organization to handle, know and be able to apply, all of these WS security related goals, requirements, architectures, mechanisms, trade-offs, etc. that are gathered across multiple and overlapping standards and specifications, in a consistent and complete fashion. The lack of a global approach and guidance that organizes and articulates all this knowledge has caused that many organizations have shown themselves very reticent to use WS-based technologies.

Concerning the definition of processes for WS secure system development, we can highlight the extension for the methodology oriented to Tropos agents and goals defined in [32]. Here, it is stated an adaptation of Tropos that lets us define the architecture that covers a certain set of requirements QoS of WS. EFSOC [33] is a event-driven framework driven for WS-based systems that defines a security model that can be easily fixed for systems in which the modifiability degree is high and therefore, they require a review and update of the authorization policies. In [34], a methodical and formal analysis based on "formal analysis of security-critical service-based software systems" is presented and in [35] a formal approach to the construction of service-based systems is presented. In [36] a MDA approach to defining a Web Services Security Architecture is described. This approach does not provide a specific and systematic method to come up with the security requirements of the system; however, PWSSec's WSSecReq subprocess provide such a method to software developers. Nagaratnam et al. explain in [37] a process that "looks at the business-application life cycle and propose a policy-driven approach overlaid on a model-driven paradigm for addressing security requirements".

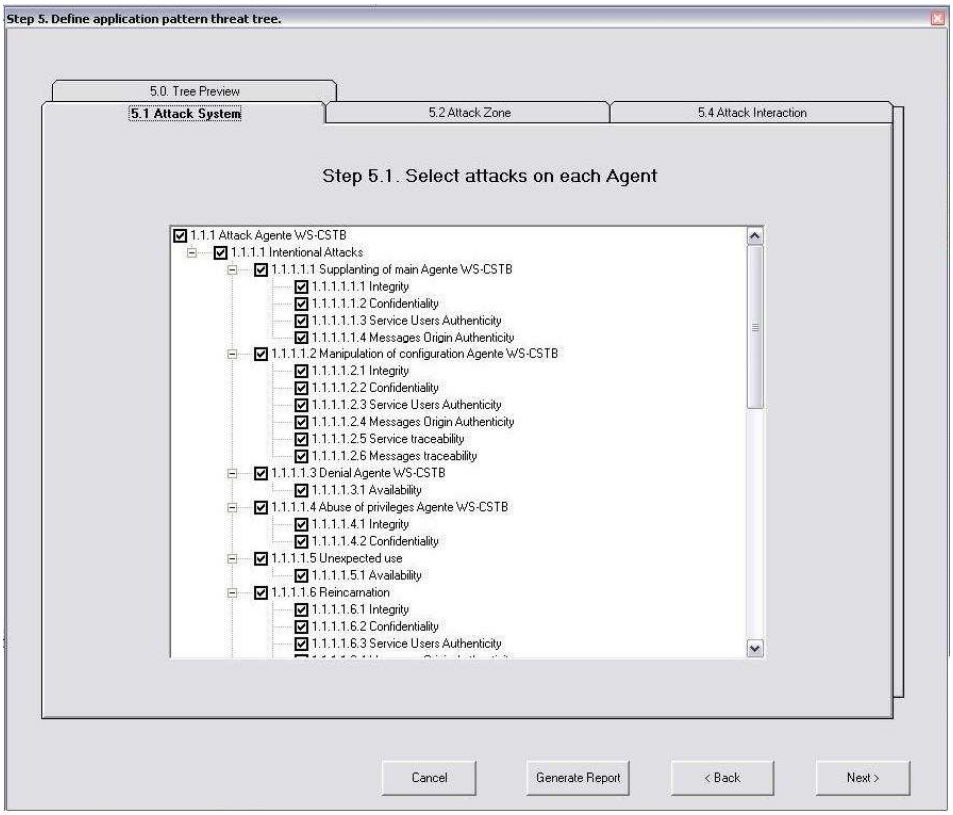

Figure 11. Tool prototype's screenshot that shows how the TA tree can be defined.

This work emphasizes deployment and managing security as complementary activities to be considered in addition to system's model, design and implementation. Although security engineering and risk analysis and management is mentioned, not a formal approach to security requirements engineering is proposed.

6. Conclusions and future research

Security is a crucial aspect if WS-based systems are to be the 'de facto' solution for inter- and intra- integrating heterogeneous systems [39].

In this article, we have presented an overview of the

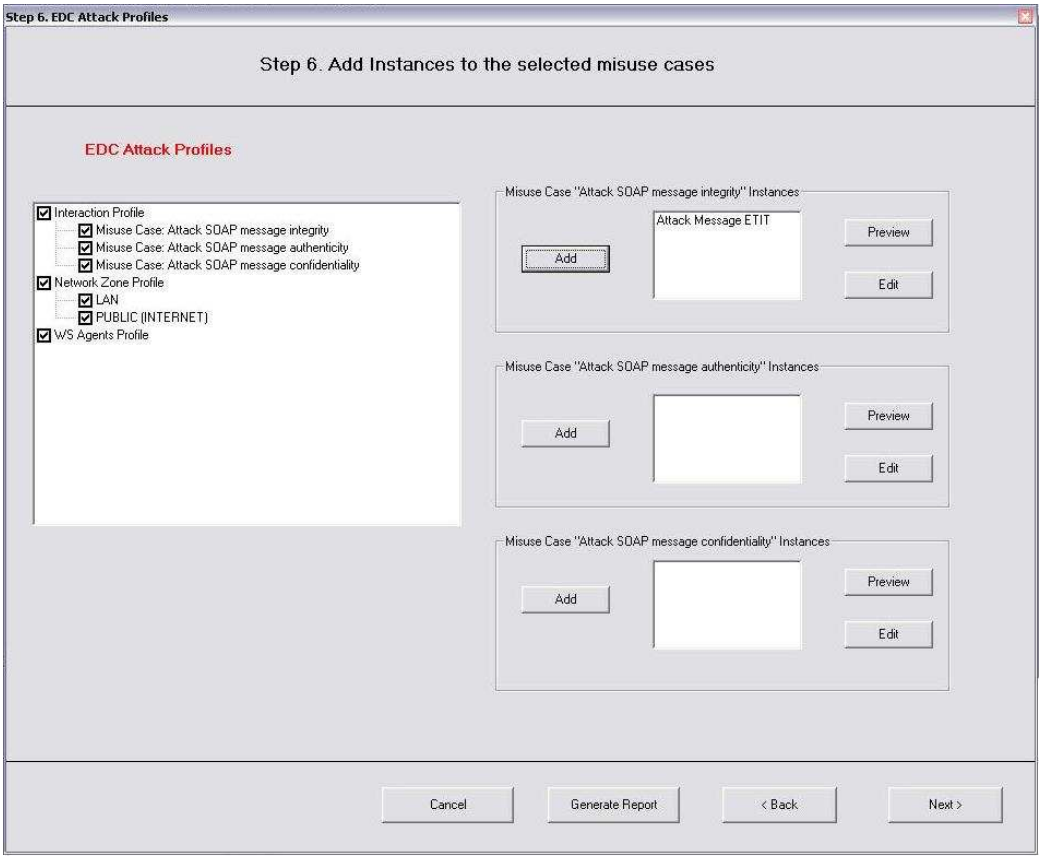

Figure 12. Dialog box that allows the aggregation of misuses cases' instances.

Breu et al. define another method for developing Web Services Security where security architectural views are explained [38]. PWSSec define security architectural views adhered to the IEEE 1471-2000 standard. In addition, this work does not provide either practical activities for gathering security requirements in WS-based systems following a security engineering approach.

None of these aforementioned approaches proposes a method such as PWSSec that, from the business goals, system security goals and functional business and application WS patterns, a WS-based secure system can be designed. Moreover, none of these methods offer facilities for the reusability of the generated products in a way that their practical applicability is guaranteed.

PWSSec process. Then, we have focused our discussion on the reusable artifacts used during the elicitation activity of the WSSecReq. The stated application of these artifacts enables developers to perform a systematic approach that will produce a complete WS-based security requirement specification. In addition, all these artifacts used during elicitation expose associations among them that provide full traceability. This traceability lets us know what security requirements have been derived from which fragment of functionality and vice-versa. This traceability connects the fragment of software functionality whose security is under analysis with the set of security requirements elicited through a set of security artifacts

(e.g. threat attack trees, misuse cases, security use cases, etc.).

Up to now, we've applied the PWSSec process in two real case studies (see [4, 40, 41]). As a consequence of its practical application we are constantly improving the process and completing the repositories with new security artifacts as those used in the WSSecReq subprocess.

In addition, a prototype tool has been developed that assists in the execution of the WSSecReq's elicitation activity. Figure 11 and 12 show two sample prototype's screenshots. In Figure 12, a screenshot where the security attack tree is instantiated from the selected WS-based business pattern and WS-based application patterns is shown. As can be seem, the tool allows the actor to select the appropriate set of threats among the set of all the possible ones. Figure 12 shows the dialog box that, in function of the TA tree, allows the instantiation of the suitable misuse cases' templates.

Finally, some of the research lines we are currently working on are listed below:

- To define and refine TA trees at the business level in order to obtain a complete security vision of the problem. This analysis is producing new business-level TA trees, attack profiles, misuse business cases, security business cases and business security requirements templates.

- To analyze the potential relationships that may exist between branches defined within and between TA trees defined at different abstraction levels (e.g. business, application, etc.).

- To define a formal meta-model for the artifacts in order to make it possible not only to create a repository of reusable artifacts but also to provide tool-based support to developers during the activity of elicitation.

- To incorporate threat and attack trees as a result of taking into account the WS-based Runtime patterns. From the abstraction point of view, WS-based runtime patterns refine WS-based application patterns.

Acknowledgments

This research is part of the following projects: DIMENSIONS (PBC-05-012-2) and MISTICO (PBC06-0082) financed by FEDER and by the "Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha" (Spain), y RETISTRUST (TIN2006-26885-E) granted by the "Dirección General de Investigación del Ministerio de Ciencia y Tecnología" (Spain).

References

[1] C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "PWSSec: Process for Web Services Security," presented at IEEE International Conference on Web Services 2006, Chicago, USA, 2006.

[2] M. Endrei, J. Ang, A. Arsanjani, S. Chua, P. Comte, P. Krogdahl, M. Luo, and T. Newling, *Patterns: Services Oriented Architectures and Web Services*, 2004.

[3] M. Endrei, J. Ang, A. Arsanjani, S. Chua, P. Comte, P. Krogdahl, M. Luo, and T. Newling, "Patterns: Service-Oriented Architecture and Web Services," *IBM Redbook*, 1st ed, 2004, pp. 345.

[4] C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "Web Services Enterprise Security Architecture: a Case Study," presented at ACM Workshop on Security on Web Services, Fairfax, Virginia, USA, 2005.

[5] C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "Web Services-based Security Requirement Elicitation," presented at 1st International Workshop on Service-Oriented Computing: Consequences for Engineering Requirements (SOCCER'05) in conjunction with IEEE RE'05, Paris, France, 2005.

[6] C. Gutiérrez, B. Moros, A. Toval, E. Fernández-Medina, and M. Piattini, "Security Requirements for Web Services based on SIREN," presented at Symposium on Requirements Engineering for Information Security, Paris, France, 2005.

[7] A. Toval, J. Nicolás, B. Moros, and F. García, "Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach," *Requirements Engineering Journal*, vol. 6, pp. 205-219, 2001.

[8] R. Breu, K. Burger, M. Hafner, J. Jürjens, G. Popp, V. Lotz, and G. Wimmel, "Key Issues of a Formally Based Process Model for Security Engineering," presented at 16th International Conference on Software and Systems Engineering and their Applications (ICSSEA'03), 2003.

[9] B. Schneier, "Attack Trees: Modeling Security Threats," *Dr. Dobb's Journal*, 1999.

[10] WS-I, "Security Challenges, Threats and Countermeasures Versión 1.0," vol. 2005: WS-I, 2005.

[11] G. Sindre and A. L. Opdahl, "Eliciting Security Requirements with Misuse Cases," presented at TOOLS-37'00, Sydney, Australia, 2000.

[12] I. Alexander, "Misuse Cases: Use Cases with Hostile Intent," *IEEE Computer Software*, vol. 20, pp. 58-66, 2003.

[13] A. P. Moore, R. J. Ellison, and R. C. Linger, "Attack Modelling for Information Security and Survivability," Software Engineering Institute 2001.

[14] OMG, "UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms," 2004.

[15] D. G. Firesmith, "Security Use Cases," *Journal of Object Technology*, vol. 2, pp. 53-64, 2003.

[16] J. Jürjens, *Secure Systems Development with UML*: Springer, 2005.

[17] D. G. Firesmith, "Engineering Security Requirements," *Journal of Object Technology*, vol. 2, pp. 53-68, 2003.

[18] D. G. Smith, "Common Concepts Underlying Safety, Security, and Survivability Engineering," SEI, Technical Note CMU/SEI-2003-TN-033, December 2003 2003.
[19] D. G. Firesmith, "Specifying Reusable Security Requirements," *Journal of Object Technology*, vol. 3, pp. 61-75, 2004.
[20] K. M. Khan and J. Han, "A Process Framework for Characterising Security Properties of Component-Based Software Systems," presented at Australian Software Engineering Conference (ASWEC'04), 2004.
[21] G. Jonsdottir, L. Davis, and R. Gamble, "Designing Secure Integration Architectures," presented at ICCBSS 2003, 2003.
[22] C. Nott, "Patterns: Using Business Service Choreography In Conjuction With An Enterprise Service Bus," 2004.
[23] J. D. Moffett, C. B. Haley, and B. Nuseibeh, "Core Security Requirements Artefacts," Open University, Department of Computing 2004/24, 2004 2004.
[24] N. R. Mead, E. D. Hough, and T. R. S. II, "Security Quality Requirements Engineering (SQUARE) Methodology," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA CMU/SEI-2005-TR-009, November 2005 2005.
[25] G. Lami, "QuARS: A Tool for Analyzing Requirements," Carnegie Mellon University. Sofware Engineering Institute., Pittsburgh, Technical Report CMU/SEI-2005-TR-014, September 2005 2005.
[26] M. E. Fagan, "Advances in Software Inspections," *IEEE Transactions on Software Engineering*, vol. 12, pp. 744 - 751, 1986.
[27] R. Shirey, "Internet Security Glossary (RFC 2828)," 2000.
[28] F. L. Crespo, M. Á. A. Gómez, J. Candau, and J. A. Mañas, "MAGERIT - Versión 2. Metodologías de Análisis y Gestión de Riesgos de los Sistemas de Información. III - Guía de Técnicas.," Ministerio de Administraciones Públicas, Madrid NIPO-326-05-047-X, 16 de Junio de 2005 2005.
[29] G. Sindre and A. L. Opdahl, "Eliciting Security Requirements with Misuse Cases," *Requirements Engineering Journal*, vol. 10, pp. 34-44, 2005.
[30] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, "W3C Web Services Description Language (WSDL) 1.1 - W3C Note 15 March 2001," 2001.
[31] G. Sindre, D. G. Firesmith, and A. L. Opdhal, "A Reuse-Based Approach to Determining Security Requirements," presented at 9th International Workshop on Requirements Engineering: Foundation of Software Quality (REFSQ'03), Klangenfurt, Velden, Austria, 2003.
[32] M. Aiello and P. Giorgini, "Applying the Tropos Methodology for Analysing Web Services Requirements and Reasoning about Qualities of Services," *UPGRADE*, vol. 5, pp. 20-26, 2004.
[33] K. Leune and M. Papazaglou, "Specification and Querying of Security Constraints in the EFSOC Framework," presented at International Conference on Service Oriented Computing, New York City, USA, Willem-Jan van den Heuvel.
[34] M. Deubler, J. Grünbauer, J. Jürjens, and G. Wimmel, "Sound Development of Secure Service-based Systems," presented at 2nd International Conference on Service Oriented Computing (ICSOC'04), New York, USA, 2004.
[35] M. Deubler, J. Grünbauer, G. Popp, G. Wimmel, and C. Salzmann, "Towards a Model-Based and Incremental Development Process for Service-Based Systems," presented at The IASTED Conference on Software Engineering (IASTED SE 2004), Innsbruck, Austria, 2004.
[36] Y. Nakamura, M. Tatsubori, T. Imamura, and K. Ono, "Model-Driven Security Based on a Web Services Security Architecture," presented at IEEE International Conference on Services Computing (SCC'05), Orlando, Florida, USA, 2005.
[37] N. Nagaratnam, A. Nadalin, M. A. Hondo, M. McIntosh, and P. Austel, "Business-driven application security: From modeling to managing secure applications," *IBM Systems Journal*, vol. 44, pp. 847-867, 2005.
[38] M. Breu, R. Breu, M. Hafner, and A. Nowak, "Web Service Engineering - Advancing a New Software Engineering," presented at 5th International Conference on Web Engineering (ICWE'05), Sydney, Australia, 2005.
[39] J. Zhang, "Trustworthy Web Services: Actions for Now," *IEEE IT Pro*, vol. 7, pp. 32-36, 2005.
[40] C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "Developing web services security systems: a case study," *International Journal of Web Engineering and Techonlogy*, vol. 2, pp. 292-306, 2006.
[41] C. Gutierrez, E. Fernández-Medina, and M. Piattini, "Security Risk Analysis in Web Services Systems," presented at International Conference on Security and Cryptography (SECRYPT 2006), Setúbal (Portugal), 2006.

Carlos Gutiérrez is PhD. by the University of Castilla-La Mancha (Spain) and MSc. by the Autonomous University of Madrid, (Spain). He has developed his professional activity in national and international companies, making consultancy activities, and in state-owned companies, as internet analyst. At the moment he is project manager in Correos Telecom (subsidiary of the Spanish state operator). His research activity is focused on web services security and secure software architectures. He has several papers in international conferences and he has published diverse articles in national and international magazines on these subjects. He is participating at the ALARCOS research group and he is an ACM member. His e-mail address is: carlos.gutierrez@correos.es.

PhD. and MSc. in Computer Science (Castilla-La Mancha and Sevilla University). He is Assistant Professor at the Escuela Superior de Informática of the Universidad de Castilla-La Mancha at Ciudad Real. His research activity is security in databases and information systems, and in multimedia documents, and he is extending his research to web services security, data warehouses security, and security metrics. He is co-editor of several books and chapter books on these

subjects, and he has several dozens of papers in national and international conferences. He participates at the ALARCOS research group of the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. He belongs to various professional and research associations (ATI, AEC, AENOR, IFIP WG11.3etc.). His e-mail is: eduardo.fdezmedina@uclm.es.

Mario Piattini is MSc and PhD in Computer Science by the Politechnical university of Madrid. Certified Information System Auditor by ISACA (Information System Audit and Control Association). Full Professor at the Escuela Superior de Informática of the Castilla-La Mancha University. Author of several books and papers on databases, software engineering and information systems. He leads the ALARCOS research group of the Departmentof Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. His research interests are: advanced database design, database quality, software metrics, object oriented metrics, software maintenance. His e-mail address is Mario.piattini@uclm.es