Available online at www.sciencedirect.com
ScienceDirect
Computer Standards & Interfaces 29 (2007) 244–253
COMPUTER STANDARDS & INTERFACES
www.elsevier.com/locate/csi

A common criteria based security requirements engineering process for the development of secure information systems

Daniel Mellado [a,*], Eduardo Fernández-Medina [b], Mario Piattini [b]

[a] Information Technology Center of the National Social Security Institute, Ministry of Labour and Social Affair, Madrid, Spain
[b] 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

Received 2 April 2006; accepted 18 April 2006
Available online 9 June 2006

Abstract

In order to develop security critical Information Systems, specifying security quality requirements is vitally important, although it is a very difficult task. Fortunately, there are several security standards, like the Common Criteria (ISO/IEC 15408), which help us handle security requirements. This article will present a Common Criteria centred and reuse-based process that deals with security requirements at the early stages of software development in a systematic and intuitive way, by providing a security resources repository as well as integrating the Common Criteria into the software lifecycle, so that it unifies the concepts of requirements engineering and security engineering.

Keywords: Common Criteria; **ISO/IEC** 15408; **ISO/IEC** 17799; Security requirement; Security requirements engineering

1. Introduction

In the last years we have observed more and more organizations becoming heavily dependent on Information Systems (**IS**). Nevertheless, software applications are increasingly ubiquitous, heterogeneous, mission-critical and vulnerable to unintentional or intentional security incidents [4,10], so that it is absolutely vital that **IS** are properly ensured from the very beginning [1,14], due to the potential losses faced by organizations that put their trust in all these **IS** and because it is cost-effective and also brings about more robust designs. Therefore, security is among the non-functional requirements which are more seriously taken into account nowadays.

However, increasing the complexity of applications and services carries out a correspondingly greater difficulty in developing security critical **IS**. In order to try to solve this problem in the last few years it has been developed a huge collection of security standards which make it easier the task of developing security critical standards. There are several standards, such as **ISO/IEC** 17799, **ISO/IEC** 13335 or **ISO/IEC** 15408, and each one helps us deal with security requirements in a way along all the **IS** development cycle. Although these standards do not give methodological support. In addition, despite of this spectacular growth there do not exist development processes that facilitate systematic treatment of security requirements within all stages of the software development lifecycle.

A very important part in the software development process for the achievement of secure software systems is that known as Security Requirements Engineering which provides techniques, methods and standards for tackling this task in the **IS** development cycle. It should involve the use of repeatable and systematic procedures in an effort to ensure that the set of requirements obtained is complete, consistent and easy to understand and analyzable by the different actors involved in the development of the system [11]. A good requirements specification document should include both functional (related to the services which the software or system should provide), and non-functional requirements (related to aspects known as features of quality, performance, portability, security, etc).

After having performed a comparative analysis of several relevant proposals of **IS** security requirements, as those of Toval et al. 2001 [20], Popp et al. 2003 [18], Firesmith 2003 [7], Breu et al. 2004 [3], etc., in Ref. [16], we concluded that those proposals did not reach the desired level of integration into the development of **IS**,

* Corresponding author.
E-mail addresses: Daniel.Mellado@alu.uclm.es (D. Mellado), Eduardo.FdezMedina@uclm.es (E. Fernández-Medina), Mario.Piattini@uclm.es (M. Piattini).

0920-5489/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2006.04.002

nor are specific enough for a systematic and intuitive treatment of **IS** security requirements at the first stages of software development. Therefore, in this article we will present the Security Requirements Engineering Process (**SREP**), which describes how to integrate security requirements into the software engineering process in a systematic and intuitive way. In order to achieve this goal, our approach is based on the integration of the Common Criteria (**CC**) (**ISO/IEC** 15408) into the software lifecycle model, which helps us specify security requirements, as well as specify the security attributes of products, and to determine if products actually meet their claims. Furthermore, we suggest evaluating the security of the **IS** along with the security engineering process by using the **CC** assurance requirements and the Systems Security Engineering Capability Maturity Model (**SSE-CMM**) at the same time, with the help of the approach of Jongsook Lee et al. (**CC_SSE-CMM**) [12]. Therefore, both standards allow us to deal with security requirements along all the **IS** development lifecycle, together with the reuse of security requirements which are compatible with the **CC** Framework subset, so that it can be assured that a security product with a high reliability will be developed by conducting a **CC**-based security development process, along with the help of the **SSE-CMM** evaluation. Moreover, **SREP** has been developed by taking into account the standard **ISO/IEC** 17799:2005, thus it conforms to the sections about security requirements of this standard (sections: 0.3, 0.4, 0.6 and 12.1). In addition, in order to support this method and make easy the treatment and specification of the security requirements, assets, security objectives and threats, we will propose the use of several concepts and techniques: a security resources repository (with assets, threats, requirements, etc), the use of **UMLSec** [18], misuse cases [19], threat/attack trees, and security uses cases [7]. These latter techniques will be used following the criteria of effectiveness, and they will allow us to integrate security aspects into an **IS** development process from the beginning, for example by expressing security-related information within the diagrams in a **UML** system specification thanks to **UMLSec**.

To describe our proposal, we will rely on the process description patterns used in the Unified Process (**UP**) [2], since it is a use-case and risk-driven, architecture-centric, iterative and incremental development process framework that leverages the Object Management Group's (**OMG**) **UML** and that is compliant with the **OMG's** Software Process Engineering Meta-model (**SPEM**). According to the **UP** the remainder of this article is set out as follows: first of all, we will briefly explain the security standards which are used by **SREP**. In Section 3, we will outline an overview of our Security Requirements Engineering Process. Section 4 will explain the activities and artifacts of **SREP**. Section 5 we will define the roles which intervene in the process. We will describe the iterations in Section 6. And, in Section 7 we will present the related work. Lastly, our conclusions and further research will be set out in Section 8.

2. Security standards

There is a highly sophisticated collection of security standards, but the most important ones regarding security requirements and which **SREP** uses are summarized as follows.

The Common Criteria (**CC**) [8] is an international standard (**ISO/IEC** 15408) for computer security. Its purpose is to allow users to specify their security requirements, to allow developers to specify the security attributes of their products, and to allow evaluators to determine if products actually meet their claims. In addition it presents requirements for the **IT** security of a product or system under the distinct categories of functional requirements and assurance requirements. The **CC** functional requirements define desired security behaviour. Assurance requirements are the basis for gaining confidence about the fact that the claimed security measures are effective and correctly implemented.

The Systems Security Engineering Capability Maturity Model (**SSE-CMM,**) (**ISO/IEC** 21827) is a model derived from the **CMM** and it describes the characteristics essential to the success of an organization's security engineering process, and it is applicable to all security engineering organizations. In contrast to the original **CMM** it defines 22 Process Areas (**PA**). The **SSE-CMM** establishes a framework for measuring and improving performance in the application of security engineering principles. It is intended to be used as a: tool for engineering organizations to evaluate their security engineering practices and define improvements to them; standard mechanism for customers to evaluate a provider's security engineering capability; and basis for security engineering evaluation organizations to establish capability based confidences.

In addition, there are proposals such as the **CC_SSE-CMM** [12] which effectively integrates the **SSE-CMM** security engineering process and the **CC**-based Target Of Evaluation (**TOE**) assurance evaluation. It consists of 23 Process Area (**PA**) and Base Practices (**BP**) per **PA**, and Generic Practices (**GP**) per capability level. It is organized into cross-mapping of **CC** assurance component to **CC_SSE-CMM PA(BP)** and **GP**, and mapping per **CC_SSE-CMM PA(BP)** and **GP** to **CC** assurance component.

The **ISO/IEC** 13335 provides guidance on the management of **IT** security and it is entitled "Information Technology — Guidelines for the management of **IT** security" (**GMITS**). It consists of five parts. Part 1, Concepts and Models, introduces a series of concepts and models for **IT** Security that are independent of the nature of the organization. Part 2, Managing and Planning **IT** Security, presents the issues that an organization must tackle before establishing or altering its **IT** Security program. Part 3, Techniques for the Management of **IT** Security, pays particular attention to the complex topic of **IT** security risk assessment; several different approaches to risk assessment are considered. Part 4, Selection of Safeguards, discusses the relative merits of different solutions and provides pointers to readily available safeguard catalogues; these catalogues are sensitive to differing national legislation. Part 5, Safeguards for External Connections, looks at the problem of crossing the "trust boundary."

The **ISO/IEC** 17799 is an information security standard published in 2005 by the **ISO/IEC**. It is entitled "Information technology — Security techniques— Code of practice for information security management". It provides best practice recommendations on information security management for use by those who are responsible for initiating, implementing or maintaining information security management systems.

Information security is defined within the standard as the preservation of confidentiality (ensuring that information is accessible only to those authorised to have access), integrity (safeguarding the accuracy and completeness of information and processing methods) and availability (ensuring that authorised users have access to information and associated assets when required).

The ISO/IEC 27001 is an information security standard published in 2005 by the ISO/IEC. Its complete name is "Information technology – Security techniques – Information security management systems – Requirements". ISO/IEC 27001:2005 specifies the requirements for establishing, implementing, operating, monitoring, reviewing, maintaining and improving a documented Information Security Management System (ISMS). It specifies requirements for the management of the implementation of security controls. The current standard is a revision of BS7799-2:2002 and it is complementary to the new standard ISO/IEC 17799:2005.

Finally, SREP uses and integrates all the former standards in different activities of the requirements engineering process. It integrates the CC security functional requirements into the elicitation of security requirements and it also introduces the CC security assurance requirements into the software quality activities. Furthermore, SREP proposes the use of the SSE-CMM (ISO/IEC 21827) in order to help in the evaluation of the security engineering process, with the help of the CC_SSE-CMM [12] approach. In addition, SREP suggests using the ISO/IEC 13335 (GMITS) to carry out the risk assessment. Moreover, SREP has been developed by taking into account the standard ISO/IEC 17799:2005, thus it conforms to the sections about security requirements of this standard (sections: 0.3, 0.4, 0.6 and 12.1) and it has also been taken into account the standard ISO/IEC 27001:2005, so that it conforms to some sections of this standard (sections: 4.2.1, 4.2.3, 4.3, 6.a, 6.b and A.12.1.1).

3. A general overview of SREP

The Security Requirements Engineering Process (**SREP**) is an asset-based and risk-driven method for the establishment of security requirements in the development of secure Information Systems and whose focus seeks to build security concepts at the early phases of the development lifecycle. Basically, this process describes how to integrate the **ISO/CC** into the software lifecycle model together with the use of a security resources repository to support reuse of security requirements (modelled with **UMLSec** [18], or expressed as security use cases or as plain text with formal specification), assets, threats (which can be expressed as misuse cases, threat/attack trees, **UMLSec** diagrams) and countermeasures.

As it is described in Fig. 1, where we show a brief outline of **SREP**, the **UP** lifecycle is divided into a sequence of phases, and each phase may include many iterations. Each iteration is like a mini-project and it may contain all the core workflows (requirements, analysis, design, implementation, and test), but with different emphasis depending on where the iteration is in the lifecycle. Moreover, the core of **SREP** is a micro-process, made up of nine activities which are repeatedly performed at each iteration throughout the iterative and incremental development, but also with different emphasis depending on what phase of the lifecycle the iteration is at. Thus, the model chosen for **SREP** is iterative and incremental, and the security requirements evolve along the lifecycle.

At the same time, **CC** Components are introduced into the software lifecycle, so that **SREP** uses different **CC** Components according to the phase and activity, although the Software Quality

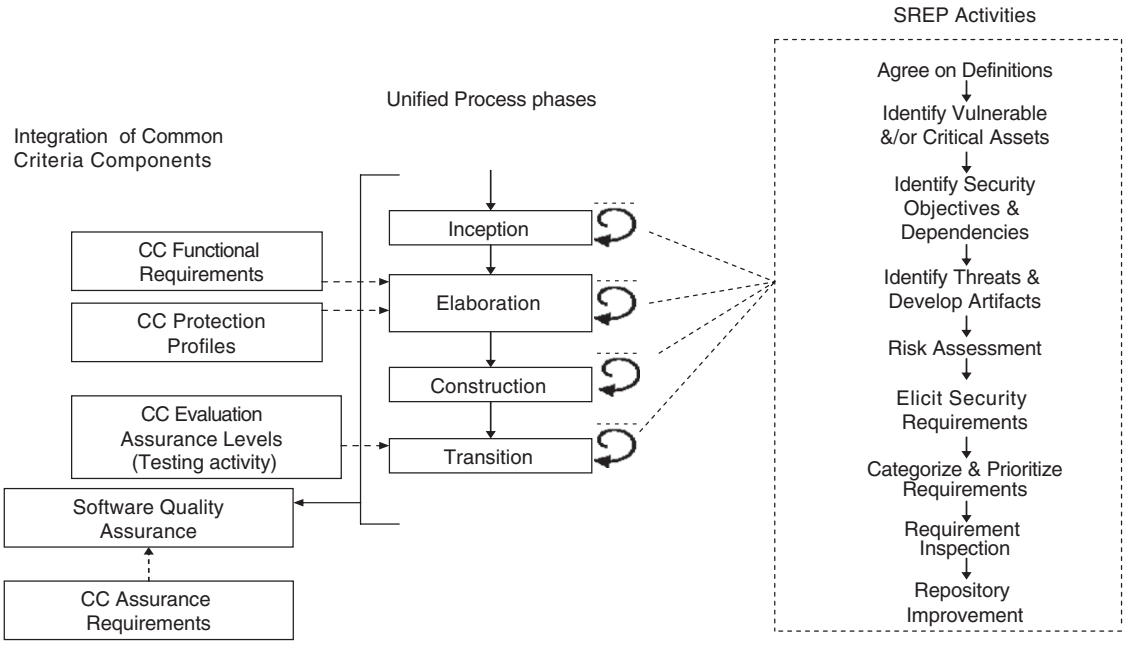

Fig. 1. The **SREP** overview.

Assurance (**SQA**) activities are performed along all the phases of the software development lifecycle. And it is in these **SQA** activities where the **CC** assurance requirements might be incorporated into, according to Kam [9]. Moreover, not only we do propose to incorporate the **CC** assurance requirements, but also we suggest that the **SSE-CMM** can be incorporated at the same time with the help of the **CC_SSE-CMM** approach [12]. Referring to **CC_SSE-CMM** Part 3 (**CC_SSE-CMM BP** and **GP** mapping to **CC** assurance component) the appropriate Base Practice and Generic Practice for each **CC** assurance requirement can be selected. Thus evaluating the **CC** assurance components it can be easily checked the related Process Areas and therefore it is parallelly evaluated the security of the **IS** and the security engineering process.

In addition, it facilitates the requirements reusability. The purpose of development with requirements reuse is to identify descriptions of systems that could be used (either totally or partially) with a minimal number of modifications, thus reducing the total effort of development [5]. Moreover, reusing security requirements helps us increase their quality: inconsistency, errors, ambiguity and other problems can be detected and corrected for an improved use in subsequent projects [20]. Thereby, it will guarantee us the fastest possible development cycles based on proven solutions.

3.1. The security resources repository

We propose a *Security Resources Repository* (**SRR**), which stores all the reusable elements. The repository, as a **SIREN** [20] approach, supports the concepts of domains and profiles. The former consists of belonging to a specific application field or functional application areas, such as e-commerce. The latter consists of a homogeneous set of requirements which can be applied to different domains, as for example personal data privacy legislation. We propose to implement the domains and profiles by taking advantage of the **CC** concepts of packages and Protection Profiles (**PP**). Thus, the requirements are stored as standardized subsets of specific security requirements together with their related elements of **SRR** (threats, etc.). In brief, each domain or profile is a view of the global **SRR**. Furthermore, the elements included in the **SRR** have been generically established by using parameter-based mechanisms, such as reusable parameterized templates. But there are also non-parameterized templates and checklists, such as asset checklists.

A meta-model, which is an extension of the meta-model for repository proposed by Sindre et al. [19], showing the organization of the **SRR** is exposed below in Fig. 2. The dark background in the objects represents our contribution to the meta-model.

As it is presented, it is an asset-driven as well as a threat-driven meta-model, because the requirements can be retrieved via assets or threats. Next, we will outline the most important and/or complex aspects of the meta-model:

- Generic Threat' and 'Generic Security Requirement' are described independently of particular domains. And they can be represented as different specifications, thanks to the elements 'Threat Specification' and 'Security Requirement Cluster Specification.
- Security Requirement Cluster' is a set of requirements that work together to satisfy the same security objective and mitigate the same threat. We agree with Sindre et al. [19] that, in many cases, it is a bigger and more effective unit of reuse.
- The 'Req–Req' relationship allows an inclusive or exclusive trace between requirements. An exclusive trace between requirements means that they are mutually alternative, as for example that they are in conflict or overlapping, whereas, an inclusive trace between requirements means that to satisfy one, another/other/s is/are needed to be satisfied.

In addition, there could have been links further on to design level specifications, security test cases, countermeasures, etc.,

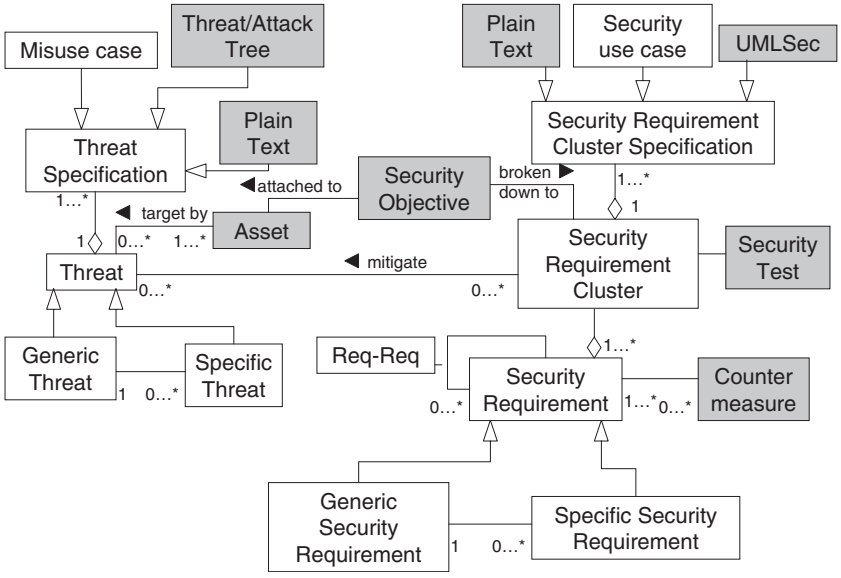

Fig. 2. The meta-model for security resources repository.

due to the fact that our proposed model process is based on the concept of iterative software construction.

Finally, we would like to point out the fact that using the **CC**, a large number of security requirements on the system itself and on the system development can be defined. Nevertheless, the **CC** does not provide us with methodological support, nor contain security evaluation criteria pertaining to administrative security measures not directly related to the **IS** security measures. However, it is known that an important part of the security of an **IS** can be often achieved through administrative measures. Therefore, according to **ISO/IEC** 17799:2005, we propose to include legal, statutory, regulatory, and contractual requirements that the organization, its trading partners, contractors, and service providers have to satisfy, and their socio-cultural environment. After converting these requirements into software and system requirements format, these requirements along with the **CC** security requirements would be the initial subset of security requirements of the **SRR.**

4. Activities and artifacts

Starting from the concept of iterative software construction of the **UP**, we will propose a micro-process, made up of nine activities which are repeatedly performed at each iteration throughout the iterative and incremental development, but with different emphasis depending on where the iteration is situated within the lifecycle, and each iteration will generate internal (or external) releases of various artifacts which altogether constitute a baseline. As the Security Requirements Specification document will evolve during the rest of the lifecycle; for instance, during design, the specification could be enriched with requirements related to the technological environment. Moreover, each security requirement can be traced along the levels of abstraction, and also, as the model understands the concepts of profiles and domains (that may be made up of elements of different abstraction level), they will be analysed by stakeholders who have the best knowledge or/and the responsibility of the domain. Furthermore, we agree with Nuseibeh [17] that the **RE** and architecture design processes are concurrent and influence each other.

The nine activities (based on [19] and [15]) that form the micro-process for the security requirements engineering, along with the external and visible artifacts that are generated within these activities, are presented below:

- *Activity 1: Agree on definitions*. The first task for the organization is to define the stakeholders and to agree upon a common set of security definitions, along with the definition of the organizational security policies and the security vision of the **IS**. It is in this activity when the *Vision Document* artifact is created and it must contain the general vision of the **IS** with a special focus on security aspects. In addition the stakeholders will participate in these latter tasks, and the candidate definitions will be mainly taken from **ISO/IEC** and **IEEE** standards, such as **ISO/IEC** 13335, **ISO/IEC** 17799:2005, **ISO/IEC** 27001:2005, **ISO/IEC** 9126, **IEEE Std**. 830:1998, or **IEEE Std**. 1061-1992.
- *Activity 2: Identify vulnerable and/or critical assets*. This is where the **SRR** is used for the first time. It consists of the identification of the different kinds of valuable or critical assets as well as vulnerable assets by the requirements engineer, who can be helped by using:
  - Lists of assets of the **SRR**, where the assets can be searched by domains, it can even be selected on a similar profile.
  - Functional requirements.
  - Interviews with stakeholders.
- *Activity 3: Identify security objectives and dependencies*. In this activity the **SRR** can be also used. Otherwise we will take into account the security policy of the Organization as well as legal requirements and other constraints in order to determine the security objectives. For each asset identified in the previous activity, the appropriate security objectives for the asset are selected and the dependencies between them are identified. Moreover the security objectives for the environment are retrieved and the assumptions about the environment are made in this activity. Security objectives are expressed by specifying the necessary security level as a probability, and they are also specified in terms of likely attacker types. The *Security Objectives Document* is developed in this activity and it may be refined in subsequent iterations (within the Inception and Elaboration phases).
- *Activity 4: Identify threats and develop artifacts*. Each asset is targeted by threat/s that can prevent the security objective from being achieved. First of all, it is necessary to find all the threats that target these assets with the help of the **SRR**. In addition, it could be necessary to develop artifacts (such as *misuse cases* or *attack trees diagrams* or **UMLSec** *use cases* and *classes* or *sequence/state diagrams*) to develop new specific or generic threats or requirements. Also it is necessary to look for threats that are not linked/related to the assets of the repository, therefore according to **CC** assurance requirements we could search in public domain sources to identify potential vulnerabilities in the **IS**, or we could instantiate the business use cases into misuse cases or instantiate the threat–attack trees associated to the business and application pattern. At this point it may be possible to take one or several existing *Protection Profiles* or *packages* and adapt them to meet modified requirements. Finally, it also defines the security problem and the conformance claims, thereby it generates the *Security Problem Definition Document* which must contain the threats, assumptions, and conformance claims. In addition, this document may be refined in subsequent iterations.
- *Activity 5: Risk assessment*. Risk must be normally determined from application to application. The final goal to achieve is the 100% risk acceptance. Firstly, it is necessary to assess whether the threats are relevant according to the security level specified by the security objectives. Then we have to estimate the security risks based on the relevant threats, their likelihood and their potential negative impacts. All of this is captured in the *Risk Assessment Document*, which is refined in subsequent iterations (within the Inception and Elaboration phases). Several methodologies can be used to carry out the

risk assessment. The **ISO/IEC** 13335 (**GMITS**), provides guidance on the use of the risk management process. In Spain it might use **MAGERIT** (the Spanish public administration risk analysis and management method) or **CRAMM** (**CCTA** Risk Analysis and Management Method) in the **UK**. Thereby, this assessment allows us to discover how the organization's risk tolerance is affected with regards to each threat. The stakeholders will take part in this activity.

- *Activity 6: Elicit security requirements*. Here, the **SRR** is used again. For each threat retrieved from the repository, one or more associated clusters of security requirements may be found. The suitable security requirements or the suitable cluster of security requirements that mitigate the threats at the necessary levels with regards to the risk assessment must be selected. However, additional requirements or clusters of requirements may be found by other means. Moreover, it might be specified the security test for each security requirement cluster, as well as an outline of the countermeasures for each security requirement, although they are refined at the design stage. Nevertheless, we agree with Firesmith [6] in the fact that care should be taken to avoid unnecessary and premature architectural mechanisms specification. Thus, at the end of this activity and according to **ISO/IEC** 17799:2005 it must have specified the functional, assurance, and organizational security requirements, along with the security requirements for the **IT** development and operational environment. Thereby, the *Security Requirements Specification Document* is created and refined in subsequent iterations.
- *Activity 7: Categorize and prioritize requirements*. Each requirement is categorized and prioritized in a qualitative ranking in a way that the most important requirements (in terms of impact and likelihood) are handled first.
- *Activity 8: Requirements inspection*. Requirements inspection is carried out in order to validate all the generated artifacts (all the documents, requirements, the modified model elements and the new generated model elements) and it is generated as a *Validation Report*. Its aim is to review the quality of the team's work and deliverables as well as assesses the security requirements engineering process. So, it is used as a sanity check. Moreover, it is verified whether the security requirements conform to the **IEEE** 830-1998 standard, because according to this standard, a requirement of quality has to be correct, unambiguous, complete, consistent, ranked for importance and/or stability, verifiable, modifiable, and traceable. After all, the security requirements documentation is written, so that a S*ecurity Requirements Rationale Document*is provided, showing that if all the security organizational, functional and assurance requirements are satisfied and all security objectives are achieved, the defined security problem is solved: all the threats are countered, the organizational security policies are enforced and all assumptions are upheld. Furthermore, it is performed within the Test workflow of the **UP** and with the help of the **CC** assurance requirements and **EALs** (Evaluation Assurance Level) and the **SSE-CMM** (**ISO/IEC** 21827). Thereby, we propose to evaluate the security of the **IS** along with the security engineering process by using the **CC** assurance requirements and the **SSE-CMM** at the same time with the help of **CC_SSE-CMM**[12]. Thus referring to **CC_SSE-CMM** Part 3, the Process Area (**PA**) in association with **CC EAL** can be selected and based on the **PA** selected it can be determined the current level of **SSE-CMM** operation capability and extract the path for the better operation capability level [12]. Thus, it can be assured that a security **IS** with a high reliability will be developed by conducting the **CC** evaluation and the **SSE-CMM** evaluation at the same time. Additionally, this activity is carried out by the quality assurer and by the inspection team at the last phase (Transition phase), with the participation of the stakeholders and security requirements engineers mainly.
- *Activity 9: Repository improvement*. The new model elements (threats, requirements, etc.) found throughout the development of the previous activities and which are considered as likely to be used in forthcoming applications and with enough quality, according to the Validation Report, are introduced into the **SRR**. Furthermore, the model elements already in the repository could be modified in order to improve their quality. Thereby, all these new or modified model elements/artifacts, which have been introduced into the **SRR**, altogether constitute a baseline. After that the *Security Target* or *Protection Profile documents* of the **CC** are written. This activity will be performed coinciding with the milestone at the end of each phase of the **UP**.

Finally, at the same time as we integrate the **CC** security functional requirements into the "Elicit security requirements" activity, we propose to outline the **EALs** in the software test

Table 1
Roles participation in SREP

X, has responsibility *, supports, O, does not participate	Business modeller	Security requirement engineer	Risk expert	Security expert	Security developer	Quality assurer	Inspection team
Agree on definitions	*	X	O	*	O	*	O
Identify vulnerable and/or critical assets	*	X	O	*	O	*	O
Identify security objectives and dependencies	*	X	O	*	O	*	O
Identify threats and develop artifacts	*	X	O	*	*	*	O
Risk assessment	O	O	X	*	O	O	O
Elicit security requirements	O	X	*	*	O	*	O
Categorize and prioritize requirements	*	X	O	*	O	*	O
Requirements inspection	*	*	*	*	*	X	X
Repository improvement	O	X	O	*	O	*	O

plan and then verify them during the transition phase. And parallelly, we propose to introduce the **CC** security assurance requirements into the software quality activities, like quality control, defect prevention and defect removal activities [9], within the Support Activities (Project Management, Configuration and Change Management) and Test workflow of the **UP**. Additionally, we suggest the use of the **SSE-CMM** (**ISO/IEC** 21827) in order to help in the evaluation of the security engineering process.

5. Roles

The roles defined here constitute a supplement to the roles in software engineering, the difference is that these roles are especially focused on security and also require special training and are based on [21]. In Table 1, it is represented the participation of each role in each activity of **SREP**:

- *Business modeller*. He/she describes the business processes, the roles involved and the artifacts produced or used in the process. He/she helps develop artifacts in **SREP** (like misuse cases, etc.) and construct the processes in a security-enhanced way, which fit in the business model of the **IS.**
- *Security requirement engineer*. This is the key role and it participates and leads most activities. It is in charge of the security vision of the **IS**, it also identifies the assets, the security objectives and its dependencies and the threats, and elicits and specifies the requirements, as well as categorizes and prioritizes the requirements with the help of other kind of specialists (if needed). Depending on the size of the project more than one person can be assigned to this role. Furthermore, this role must not necessarily have a thorough technical understanding of security, although a sound security management is required.
- *Risk expert*. This is the specialized role in security related risks and the main task of this role is to perform the security risk assessment. Additional training in security of **IS** is recommended.
- *Security expert*. The main task of the security expert is to improve the overall security of the **IS**. This role is the technical expert in security so that he/she acts as a consultant, and helps us find security relevant information, estimate the degree to which **IS** meets its security claims and define the security vision of the **IS** and the organizational security policies and measures.
- *Security developer*. The role of the security developer is to support the construction of tests to help the Requirements Inspection activity during the Test workflow of the **UP**.
- *Quality assurer*. This is the role responsible for the Requirements Inspection activity within the Test workflow of the **UP** and it could take advantage of the use of the **CC** assurance classes. In addition, this role can help us with informal reviews of the quality of the most important artifacts in each activity.
- *Inspection team*. It is a group external to the **IS** development team whose aim is to review the quality of the development team's work and deliverables as well as evaluate the security engineering process by using the **CC** assurance requirements and the **SSE-CMM**, with the help of **CC_SSE-CMM** [12]. Besides it is the role responsible for the Requirements Inspection activity within the Transition phase of the **UP**. Additionally, this team is in charge of the assurance that the **IS** meets its security claims with the help of the **EALs**.

6. Iterations

We propose an iterative and incremental security requirements engineering process, so that each iteration coincides with an iteration within a phase of the **UP**. This is because the **UP** lifecycle is divided into a sequence of phases, which may include many iterations, and each one concludes with a major milestone. This philosophy lets us take into account changing requirements, facilitates reuse and correct errors over several iterations, risks are discovered and mitigated earlier, and the process itself can be improved and refined along the way. Therefore, the result is a more robust **IS**.

The integration of SREP, with the CC and with the phases of the UP is presented below:

- *Inception*. It is the first phase and it is focused on the earlier activities of **SREP**. The security vision document is produced, and around the 50% of the first order requirements are defined, therefore a similar percentage of the assets, security objectives and threats. In addition, the security problem definition is carried out and an overall risk outline is performed. Moreover, the main focus with regard to the **CC** assurance classes is on the following classes: Composition, Lifecycle Support and Vulnerability Assessment. Also, at this point, it may be possible to take an existing or several Protection Profiles or packages and adapt them to meet modified requirements. Nevertheless, it is difficult to conduct everything in one iteration, so it might be necessary another iteration with more mature understanding of the **IS**.
- *Elaboration*. More than one iteration may be normally made at this phase depending on the size and complexity of the project. The goal of this phase, and according to **ISO/IEC** 17799:2005, is to identify around 98% of the critical/vulnerable assets, security objectives, threats and first ordered requirements and around 90% of second ordered requirements. Moreover a refinement of the risk assessment and the security problem definition is carried out. In addition, this phase is also focused on the requirements categorization and prioritization, and on the requirements inspection as well as on the security requirements rationale. Therefore, the most important **CC** assurance classes for this phase are: Security Target Evaluation, Protection Profile Evaluation, Guidance Documents, Development, and Vulnerability Assessment.
- *Construction*. At this phase, the remaining requirements are defined along with the final design and the implementation of the security countermeasures. The Requirements Inspection activity is emphasized at this phase. The main focus with regard to the **CC** assurance classes is on the following

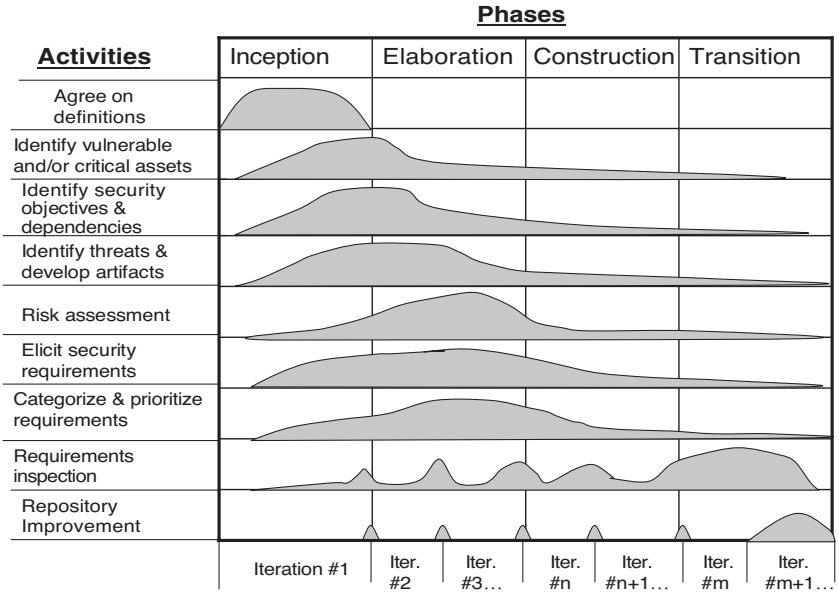

Fig. 3. The work amount per SREP iteration.

classes: Security Target or **PP** Evaluation, Development, Composition and Vulnerability Assessment.

- *Transition*. It is the last phase and when the **IS** is put into productive use. The danger is, however, that other requirements can emerge, thus security risks must be considered and therefore they must be dealt with carefully and in a pragmatic way. This phase is focused on the Requirements Inspection and Repository Improvement activities. So, the most important **CC** assurance classes for this phase are: Security Target or **PP** Evaluation, Tests, Guidance Documents, Composition, and Vulnerability Assessment.

Finally, we can see in the Fig. 3 that the core of **SREP** is performed at the earlier phases, therefore in the earlier iterations, although at later phases requirements are reviewed and it may be necessary to introduce new ones that turn up during the **IS** development process.

7. Related work

Extensive work has been carried out on security requirements during the last few years, and there are several works that deals with security requirements at the early stages of the development lifecycle, the same as **SREP**. Next, there are summarized those proposals particularly close in topic to ours and parallelly it is also explained their relation to **SREP**.

SQUARE (Security Quality Requirements Engineering Methodology) [15] is a model made up of nine steps in which it is provided a means for eliciting, categorizing and prioritizing security requirements for information technology systems and applications. **SREP** activities are based on these steps of **SQUARE** partially. However, in contrast to **SREP**, **SQUARE** does not incorporate into its steps the **CC** nor the **SSE-CMM**, and it does not make any reference to be in compliance with any Information Security Management System standard, such as **ISO/IEC** 17799 or **ISO/IEC** 27001, as well as the steps of **SQUARE** do not deal with the security requirements reuse.

The "Security-critical system development with extended use cases" approach of Popp et al. [18] suggested a methodology to integrate security aspects from the beginning into a system development process. They provide an extension to the conventional process of developing use-case-oriented process for security-critical systems. They consider security aspects both in the static domain model and in the functional specification. For the elaboration of the functional aspects they introduced a question catalogue and for the domain model an UML-extension, UMLSec. This technique, UMLSec, can be used in SREP in order to model security requirements. Although SREP, in contrast to this approach, provides a reuse repository and integrates the CC and the SSE-CMM within the IS development lifecycle.

The **SIREN** (SImple REuse of software requiremeNts) approach by Toval et al.[20], suggests a method to elicit and specify the security system and software requirements including a repository of security requirements initially populated by using **MAGERIT** and which can be structured according to domains and profiles in a similar way to **SREP**. Although **SIREN** focuses on requirements lists and it only reuses requirements, which are retrieved via **MAGERIT** asset hierarchy or via the aforementioned repository structure. A distinguishing property of our suggestion is that we suggest to reuse specifications of requirements and threats, as well as security objectives, assets, countermeasures and tests, so that the requirements can be retrieved via assets, security objectives or threats. Finally, **SIREN** is based on a spiral model whereas our approach is based on the concept of iterative software construction of the Unified Process, which is a use-case and risk-driven, architecture-centric, iterative and incremental development process framework that leverages the **OMG**.

The "Reuse-Based Approach to Determining Security Requirements" by Sindre et al. [19], proposes a reuse-based approach to determining security requirements, so that it involves several steps in order to develop with reuse, and **SREP** is based on some of these steps, adapting them to incorporate the **CC** and the **SSE-CMM**. Furthermore, it suggests a reuse repository which is the base of the security resources repository of **SREP**, although we add several objects to their meta-model, such as security objectives, tests, countermeasures, as well as we allow the specification of the requirements and threats using several techniques. However, Sindre et al. approach is only focused on the activities directly related to reuse, while **SREP** deals with all tasks concerning to security requirements elicitation and specification.

The "Holistic security requirement engineering" approach of Zuccato [21] meant to elicit security requirements according to system-theoretic considerations. It shows that security requirements can be defined with the help of investigations in the business environment, workshops with stakeholders and risk analysis. This multidimensional approach leads to a holistic understanding of the requirements that fit into the system development lifecycle. On the basis of the new definition of a holistic security requirement it is proposed a process, which relies on the process description patterns used in the Unified Process, the same as **SREP**. So three different source groups were taken into account in order to find a holistic set of requirements: activities, artifacts and roles. The roles of **SREP** are based on these roles defined by Zuccato, as well as the process description structure because both proposals rely on the process description patterns used in the Unified Process. But **SREP**, in contrast to this approach, also integrates the **CC** and the **SSE-CMM** within the **IS** development lifecycle.

In brief, the main differences between our proposal and earlier ones are as follows:

- **SREP** is a standard-based process. A **CC**-centred process which also integrates the **SSE-CMM** into the development lifecycle thanks to the **CC_SSE-CMM** approach [12] and which conforms to **ISO/IEC** 17799:2005 with regard to security requirements management.
- It is a reuse-based approach based on a security resources repository, so that they are reused threats and requirements and their specifications, security objectives, assets, countermeasures and tests.
- It is conducted by actives or threats and the risk.
- **SREP** is based on the concept of iterative software construction of the Unified Process.
- It integrates the latest security requirements specification techniques (such as **UMLSec** [18], security use cases [7] and misuse cases [19]).

8. Conclusions and further research

In our present so-called Information Society the increasingly crucial nature of **IS** with corresponding levels of new legal and governmental requirements is obvious. For this reason, the development of more and more sophisticated approaches to ensuring the security of information is becoming a need. Information Security is usually only tackled from a technical viewpoint at the implementation stage, even though it is an important aspect, but we believe it is fundamental to deal with security at all stages of **IS** development, especially in the establishment of security requirements, since these form the basis for the achievement of a robust **IS**. In fact, extending Requirements Engineering modelling and formal analysis methodologies to cope with Security Requirements has been a major effort in the past decade [13]. However, developing security critical **IS** is very difficult in part because security cannot simply be tested, but has to be ensured during the whole development process. Fortunately there are several security standards, like the Common Criteria (**ISO/IEC** 15408), which helps us deal with the security requirements along all the **IS** development cycle, although it does not give methodological support.

Consequently, the contribution of this work is that of providing a standard-based process that deals with the security requirements at the early stages of software development in a systematic and intuitive way, which is based on the reuse of security requirements, by providing a Security Resources Repository (**SRR**), together with the integration of the Common Criteria (**ISO/IEC** 15408), and also the **SSE-CMM** (**ISO/IEC** 21827) thanks to **CC_SSE-CMM** approach [12], into software lifecycle model. Furthermore, it also conforms to **ISO/IEC** 17799:2005 with regard to security requirements (sections: 0.3, 0.4, 0.6 and 12.1). In addition, starting from the concept of iterative software construction, we propose a micro-process for the security requirements engineering, made up of nine activities, which are repeatedly performed at each iteration throughout the iterative and incremental development, but with different emphasis depending on where the iteration is in the lifecycle. Finally, one of the most relevant aspects is the fact that this proposal integrates other approaches, such as **SIREN** [20], **UMLSec** [18], security use cases [7] or misuse cases [19].

Further work is also needed to provide a **CARE** (Computer-Aided Requirements Engineering) tool which supports the process, as well as a refinement of the theoretical approach by proving it with a real case study in order to complete and detail more **SREP**.

Acknowledgments

This article has been produced in the context of the **DIMENSIONS** (**PBC**-05-012-2) Project of the Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha along with the **FEDER** and the **CALIPO** (**TIC**2003-07804-**CO**5-03) and **RETISTIC** (**TIC**2002-12487-**E**) projects of the Dirección General de Investigación del Ministerio de Ciencia y Tecnología.

References

[1] R. Baskerville, The development duality of information systems security, Journal of Management Systems 4 (1) (1992) 1–12.
[2] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Software Development Process, ed. Addison-Wesley. 1999.

[3] R. Breu, K. Burger, M. Hafner, G. Popp, Towards a Systematic Development of Secure Systems, Proceedings WOSIS 2004, 2004, pp. 1–12.
[4] CERT, http://www.cert.org.
[5] J. Cybulsky, K. Reed, Requirements Classification and Reuse: Crossing Domains Boundaries, ICSR'2000, 2000, pp. 190–210.
[6] D.G. Firesmith, Engineering security requirements, Journal of Object Technology 2 (1) (2003) 53–68.
[7] D.G. Firesmith, Security use cases, Journal of Object Technology (2003) 53–64.
[8] ISO/IEC_JTC1/SC27, Information technology — Security techniques — Evaluation criteria for IT security, ISO/IEC 15408:2005 (Common Criteria v3.0). 2005.
[9] S.H. Kam, Integrating the Common Criteria Into the Software Engineering Lifecycle, IDEAS'05, 2005, pp. 267–273.
[10] R. Kemmerer, Cybersecurity, Proc. ICSE'03-25th Intl. Conf. on Software engineering, 2003, pp. 705–715.
[11] G. Kotonya, I. Sommerville, Requirements Engineering Process and Techniques, Hardcover ed., 1998, 1998, p. 294.
[12] J. Lee, J. Lee, S. Lee, B. Choi, A CC-based Security Engineering Process Evaluation Model, 27th Annual International Computer Software and Applications Conference (COMPSAC'03), 2003, p. 130.
[13] F. Massacci, M. Prest, N. Zannone, Using a security requirements engineering methodology in practice: the compliance with the Italian data protection legislation, Computers Standards and Interfaces 27 (2005) 445–455.
[14] J. McDermott, C. Fox, Using Abuse Case Models for Security Requirements Analysis, Annual Computer Security Applications Conference, Phoenix, Arizona, 1999.
[15] N.R. Mead, T. Stehney, Security Quality Requirements Engineering **(SQUARE)** Methodology, Software Engineering for Secure Systems (**SESS**05), **ICSE** 2005 International Workshop on Requirements for High Assurance Systems, St. Louis, 2005.
[16] D. Mellado, E. Fernández-Medina, M. Piattini, A Comparative Study of Proposals for Establishing Security Requirements for the Development of Secure Information Systems, The 2006 International Conference on Computational Science and its Applications (**ICCSA** 2006), Springer **LNCS** 3982 (2006) 1044–1053.
[17] D. Nuseibeh, Weaving together requirements and architectures, **IEEE** Computer (2001) 115–117.
[18] G. Popp, J. Jürjens, G. Wimmel, R. Breu, Security-Critical System Development with Extended Use Cases, 10th Asia-Pacific Software Engineering Conference, 2003, pp. 478–487.
[19] G. Sindre, D.G. Firesmith, A.L. Opdahl, A Reuse-Based Approach to Determining Security Requirements, Proc. 9th International Workshop on Requirements Engineering: Foundation for Software Quality (**REFSQ**'03), 2003, Austria.
[20] A. Toval, J. Nicolás, B. Moros, F. García, Requirements reuse for improving information systems security: a practitioner's approach, Requirements Engineering Journal (2001) 205–219.
[21] A. Zuccato, Holistic security requirement engineering for electronic commerce, Computers and Security, 23 (2004) 63–76.

Daniel Mellado is MSc in Computer Science form the Autonomous University of Madrid (Spain), and a PhD student at the Escuela Superior de Informática of the Castilla- La Mancha University (Spain). He is civil servant in the Information Technology Centre of the National Social Security Institute (in Madrid, Spain), where he works as IT analyst. His research activities are security requirements engineering, security in information systems and secure software process improvement. He is author of several papers on security requirements and improvement of secure software development process.

Eduardo Fernández-Medina is PhD and MSc in Computer Science. He is Assistant Professor at the Escuela Superior de Informática of the Universidad de Castilla- La Mancha at Ciudad Real (Spain). His research activities are security requirements, security in databases, data warehouses, web services and information systems, and also in security metrics. He is the co-editor of several books and chapter books on these subjects, and 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, WG11.3, etc).

Mario Piattini is MSc and PhD in Computer Science from the Politechnical University of Madrid. He is certified information system auditor by ISACA (Information System Audit and Control Association). He is Associate Professor at the Escuela Superior de Informática of the Castilla- La Mancha University (Spain). He is author of several books and papers on databases, security, software engineering and information systems. He leads the ALARCOS research group of the Department of 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 and software maintenance.