ALEXSSANDER ANTONIO SIQUEIRA

A PROCESS LINE PROPOSAL FOR SECURE SOFTWARE DEVELOPMENT

Thesis presented to the Postgraduate Program in Computer Science at the Pontifical Catholic University of Paraná as a partial requirement to obtain the title of Master of Science.

Curitiba
2016

ALEXSSANDER ANTONIO SIQUEIRA

A PROCESS LINE PROPOSAL FOR SECURE SOFTWARE DEVELOPMENT

Thesis presented to the Postgraduate Program in Computer Science at the Pontifical Catholic University of Paraná as a partial requirement to obtain the title of Master of Science.

Field Area: Software Engineering

Supervisor: Dra. Sheila Reinehr
Co-supervisor: Dra. Andreia Malucelli

Curitiba
2016

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor, com anuência de seu orientador.

Curitiba, 09 de dezembro de 2016.

Assinatura do Autor

Assinatura do Orientador

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central

S618p 2016

Siqueira, Alexssander Antonio
A process line proposal for secure software development / Alexssander Antonio Siqueira; [orientadora, Sheila Reinehr; co-orientador, Andreia Malucelli]. -- 2016
83 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2016
Bibliografia: f. 74-79

1. Informática. 2. Software – Desenvolvimento. 3. Software – Confiabilidade. 4. Engenharia de software. 5. Computadores - Medidas de segurança. I. Reinehr, Sheila dos Santos. II. Malucelli, Andreia. III. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. IV. Título.

CDD 20. ed. – 004.068

DEDICATION

To my endless love, my sweet wife Dra. Thalyta Marina Benetti.

ACKNOWLEDGEMENTS

This research project is made possible through the help and support from everyone, including: parents, teachers, family, friends, and LEPS colleagues.

First and foremost, I would like to thank Dra. Sheila Reinehr and Dra. Andreia Malucelli for their most support and encouragement. They accepted the challenge, they believed in my potential.

Second, I would like to thank my colleagues Regina Albuquerque and Everson Mauda that helped me to become more confident.

Finally, I sincerely thank to my wife Thalyta, my father Edgar and my two mothers Hilda and Flora. The product of this research paper would not be possible without all of them.

I do not measure a man's success by how high he climbs but how high he bounces when he hits bottom.

- George S. Patton

ABSTRACT

The secure software development is an important field of study in the Software Engineering area. To avoid a successful malicious user attack that can impact the software execution, it is necessary to introduce security requirements in the project development scope since the early stages. The secure software development process results in the security information aspects integration that requires process tailoring and management. Current practices provide guidance on secure software development process definition. However, there is a lack of methods and approaches to support the integration of security activities in the current organizations software life cycles. The concept of Software Process Lines (SPrL) can be an alternative to support, organize, manage and control the several number of extended secure development processes that demand an efficient process tailoring approach. The SPrL or family of processes consists in a set of processes built from a series of shared process assets in order to reuse process knowledge across projects with different needs. The main objective of this research project is the definition of a SPrL for secure software development. To achieve it, the research project was developed using the Action-Research method as reference. Firstly, the main secure development processes were identified and their main method contents were selected. Then, the processes common elements and variation points were analyzed and described. Next, a SPrL proposal was defined and employed in a controlled environment to be instantiated in a set of real projects. Finally, the SPrL usability and utility factors were evaluated using a SERVQUAL questionnaire answered by the process users. The proposed SPrL is a contribution for organizations that aim to develop secure applications considering the challenges of tailoring security aspects with software engineering practices.

Keywords: Software Process Line, Secure Development, Information Security

SUMMARY

ABSTRACT ..... **VII**

LIST OF FIGURES ..... **XI**

LIST OF TABLES ..... **XII**

LIST OF ABREVIATIONS ..... **XIII**

CHAPTER 1 - INTRODUCTION **1**

1.1 MOTIVATION 4
1.2 OBJECTIVES 4
1.3 SCOPE DELIMITATION 5
1.4 WORK STRUCTURE 5
1.5 DISSERTATION DOCUMENT STRUCTURE 5
1.6 CHAPTER CONSIDERATIONS 6

CHAPTER 2 - LITERATURE REVIEW **7**

2.1 SOFTWARE DEVELOPMENT PROCESS 7
2.2 SOFTWARE PRODUCT LINE 8
2.3 SOFTWARE PROCESS LINE 11
2.4 INFORMATION SECURITY 14
2.4.1 Security Definitions 15
2.4.2 Risk Management 16
2.4.3 Risk Assessment 19
2.4.4 ISO/IEC 27001 - Information Security Management Standard 21
2.5 SECURE SOFTWARE DEVELOPMENT PROCESS 23
2.5.1 ISO/IEC 15408 – Common Criteria 24
2.5.2 Microsoft SDL 25
2.5.3 OWASP CLASP 26
2.5.4 McGraw Touchpoints 27
2.5.5 ISO/IEC 21827 – SSE - CMM 27
2.5.6 ISO/IEC 27034 – Application Security 29
2.6 RELATED WORKS 37
2.6.1 Research questions definition 38
2.6.2 Searching process 38

2.6.3 Data analysis 38
2.7 CHAPTER CONSIDERATIONS 39

CHAPTER 3 - RESEARCH STRUCTURE 40

3.1. RELEVANT CONCEPTS ABOUT RESEARCH METHODOLOGY 40
3.2. RESEARCH CHARACTERIZATION 41
3.3. RESEARCH STRATEGY 42
3.3.1. Stage 1 – Define the context and stablish the research purpose 43
3.3.2. Stage 2 - Analyze the organization process development scenario 44
3.3.3. Stage 3 - Define the proposed SPrL for secure development 44
3.3.4. Stage 4 - Prepare the action plan 45
3.3.5. Stage 5 - Implement the SPrL for secure development 45
3.3.6. Stage 6 - Evaluate the SPrL implementation 46
3.4. CHAPTER CONSIDERATIONS 46

CHAPTER 4 - RESEARCH DEVELOPMENT 47

4.1. PRELIMINARY PHASE: CONTEXT AND PURPOSE 47
4.2. FIRST IMPROVEMENT CYCLE 49
4.2.1. DATA GATHERING, ANALYSIS AND FEEDBACK 49
4.2.2. ACTION PLANNING 50
4.2.3. IMPLEMENT THE SPRL FOR SECURE DEVELOPMENT 52
4.2.4. EVALUATE THE SPRL IMPLEMENTATION 60
4.3. SECOND IMPROVEMENT CYCLE 61
4.3.1. DATA GATHERING, ANALYSIS AND FEEDBACK 61
4.3.2. ACTION PLANNING 62
4.3.3. REVIEW THE SPRL FOR SECURE DEVELOPMENT 63
4.3.4. EVALUATE THE SPRL IMPLEMENTATION 64

CHAPTER 5 - RESEARCH EVALUATION 68

5.1. SERVQUAL QUESTIONNAIRE 68
5.2. SERVQUAL ANSWERS 69

CHAPTER 6 - CONCLUSION 72

7.1. RELEVANCE OF THE STUDY 72
7.2. RESEARCH CONTRIBUTION 72
7.3. RESEARCH LIMITATIONS 72
7.4. FUTURE WORKS 73

REFERENCES **74**

APPENDICES A – SERVQUAL QUESTIONNAIRE **80**

APPENDICES B – SPRL TASKS SPECIFICATION **82**

APPENDICES C – PROJECTS PLANNING **83**

LIST OF FIGURES

Figure 2-1. Software product line engineering framework (POHL; METZGER, 2006). 9
Figure 2-2. Roles in a software product line (MCGREGOR, 2004). 10
Figure 2-3. Orthogonal Variability Model (POHL; METZGER, 2006). 11
Figure 2-4. SPrL Architecture (ALEGRÍA; BASTARRICA, 2012). 13
Figure 2-5. Security concepts relationship (HARRIS, 2008). 16
Figure 2-6. Risk management process (NIST, 2011). 17
Figure 2-7. NIST definition of Risk Management (NIST, 2010). 18
Figure 2-8. Risk assessment process (NIST, 2012). 20
Figure 2-9. Risk level matrix (STEWART, 2009). 20
Figure 2-10. PDCA model applied to ISMS processes (ISO/IEC, 2013). 22
Figure 2-11. Application Security Management Process (ISO/IEC, 2011). 31
Figure 2-12. Organization Normative Framework (ISO/IEC, 2011). 33
Figure 2-13. Project impact by the use of ISO/IEC 27034 (ISO/IEC, 2011). 35
Figure 2-14. Application Security Control Library (ISO/IEC, 2011). 36
Figure 2-15. Application Security Control (ISO/IEC, 2011). 37
Figure 3-1. The action research types of steps (COUGHLAN; COGHLAN, 2002). 42
Figure 4-1. SPrL phases workflow (Author). 53
Figure 4-2. Management phase overview (Author). 54
Figure 4-3. Training phase overview (Author). 54
Figure 4-4. Requirements phase overview (Author). 55
Figure 4-5. Design phase overview (Author). 57
Figure 4-6. Implementation phase overview (Author). 58
Figure 4-7. Verification phase overview (Author). 59
Figure 4-8. Release phase overview (Author). 60
Figure 4-9. Setting tasks frequency in the EPF tool (Author). 62
Figure 4-10. EPF tool support for MS Project format export (Author). 63
Figure 4-11. SPrL organization in iterations (Author). 64
Figure 4-12. Waterfall life-cycle representation for SPrL (Author). 64
Figure 4-13. Project 01 planning with Training and Requirement phases (Author). 65
Figure 4-14. Variation point options in the Requirements phase (Author). 65
Figure 4-15. Variation point option in the Design phase (Author). 66
Figure 4-16. Project 01's Implementation and Evaluation phases planning (Author). 67
Figure 5-1. SPrL graphical overview (Author). 70

LIST OF TABLES

Table 2-1. Variability types for process lines (TERNITE, 2009). 12
Table 2-2. SSE-CMM Process Areas (ISO/IEC, 2008b) 28
Table 2-3. SSE-CMM Capabilities Levels (ISO/IEC, 2008b) 29
Table 4-1. Original process roles and artifacts relationship (Author). 48
Table 4-2. Resulting analysis from processes commonalities and variabilities (Author). 50
Table 4-3. SmatySPEM stereotypes (JUNIOR et al., 2011). 51
Table 4.4. Selected projects to the research evaluation (Author). 61
Table 5-1. Score average for each evaluation factor. 69

LIST OF ABREVIATIONS

ANF	Application Normative Framework
ASL	Application Security Level
ASC	Application Security Control
ASMP	Application Security Management Process
ASRM	Application Security Risk Management
CASPER	Context Adaptable Software Process Engineering
CC	Common Criteria
CLASP	Comprehensive, Lightweight Application Security Process
CMMI	Capability Maturity Model Integration
COBIT	Control Objectives for Information and related Technology
EBSE	Evidence-based Software Engineering
IEC	International Electrotechnical Commission
IS	Information Security
ISMS	Information Security Management System
ISO	International Organization for Standardization
ITIL	Information Technology Infrastructure Library
NIST	National Institute of Standard and Technologies
ONF	Organization Normative Framework
OMG	Object Management Group
OVM	Orthogonal Variability Management
OWASP	Open Web Application Security Project
PA	Process Area
PDCA	Plan-Do-Check-Act
SDL	Secure Development Lifecycle

SLR	Systematic Literature Review
SPEG	Software Process Engineering Group
SPEM	Software and Systems Process Engineering Meta-model Specification
SPL	Software Product Line
SPLE	Software Product Line Engineering
SPrL	Software Process Line
SPrLA	Software Process Line Architecture
SSE-CMM	Security Software Engineering – Capability Maturity Model
UML	Unified Modeling Language
VM	Variability Management

CHAPTER 1 - INTRODUCTION

We shall defend our island, whatever the cost may be, we shall fight on the beaches, we shall fight on the landing grounds, we shall fight in the fields and in the streets, we shall fight in the hills; we shall never surrender.

- *Winston Churchill*

Our world is increasingly relying on complex software and systems. In a several number of fields and industries such as transportation, finance, banking, telecommunications, medical devices, they now play a critical role and require high assurance: any failure caused by an attacker could lead to catastrophic loss in terms of cost, reputation, environment damage, or even human life (PONSARD et al., 2007).

Secure software development is an engineering area that allows the development of software systems that can avoid malicious users to attack these systems using harmful software technologies that can affect their features operation (EL-ATTAR, 2012).

The development of secure software systems is a challenge, due to errors and misspecifications in requirements, design, implementation and test that can bring vulnerabilities to the system. Vulnerability can be any weaknesses in the software that attackers can exploit to compromise the system operation (ELAHI; YU; ZANNONE, 2010).

There are frameworks, best practices and standards to support organizations in assessing their security risks, stablishing their security management system, implementing the appropriate security controls, complying with governance requirements and security regulations. However, the security attacks techniques are in frequent evolution and becoming more sophisticated. Information security must be always aware to avoid new threats, along with the available methods, techniques, policies, guidelines, educational and training approaches and technologies used to combat them (FUTCHER; SOLMS, 2008).

Secure development has two large acceptable approaches. The first approach consists in implementing security in a reactive manner, where the security aspects are

integrated in the produced software after its development. However, this approach is costly due to the effort to fix a security defect that can require a rework in terms of analysis, design, coding and testing. The second approach addresses security as a proactive process, where the security aspects are largely integrated in the development life cycle (KHAN; MUSTAFA, 2009).

The security standard ISO/IEC 15408 (Common Criteria for Information Technology Security Evaluation) provides a common set of security requirements for IT products. In addition, the standard provides the necessary guidance for assurance and evaluation of the security requirements employment (ISO/IEC, 2009). The standard ISO/IEC 15408 can be combined with the standard ISO/IEC 12207 (Systems and software engineering - Software life cycle processes) that establishes a common framework for software life cycle processes with a well-defined terminology, that can be referenced by the software industry (ISO/IEC, 2008a). The resulted combination is a set of secure development processes.

The main recognized processes in the secure development field are the Microsoft's Security Development Life Cycle[1] (SDL), OWASP's Comprehensive, Lightweight Application Security Process[2] (CLASP) and McGraw' Touchpoints[3]. All of them provide a set of integrated security activities into the development life cycle. These processes were submitted to an extensive validation, due to their use in several projects. These processes suggest the integration of some security engineering tasks and artifacts such as security requirements analysis, threat modeling, risk assessment and penetration test into the software development life cycle (WIN et al., 2009).

The maturity of a secure development process can be evaluated using the standard ISO/IEC 21827:2008 that specifies the Systems Security Engineering - Capability Maturity Model® (SSE-CMM®), which describes the essential process management characteristics and security engineering process activities that must exist to ensure the security engineering practice in an organization (ISO/IEC, 2008b). The ISO/IEC 21827:2008 does not prescribe a particular process or sequence, but captures practices generally observed in industry. This standard has a relationship with the ISO/IEC 15504 (Process Assessment), as both are dedicated to process improvement and capability maturity assessment. However, the standard ISO/IEC

[1] Available at: http://www.microsoft.com/en-us/sdl/
[2] Available at: https://www.owasp.org/index.php/CLASP_Concepts
[3] Available at: http://www.cigital.com/sdlc1/

15504 is deeply focused on software process and the ISO/IEC 21827 is focused on security engineering practices such as risk management, risk assessment and risk analysis (ISO/IEC, 2008b).

A specific standard for secure software development was released in 2011. The ISO/IEC 27034 that provides guidance to assist organizations in the integration of security aspects into their software development processes. This standard is flexible, being applicable to in-house developed applications, applications acquired from third parties, and where the development or the operation of the application is outsourced (ISO/IEC 2011).

The standard ISO/IEC 27034 supports the secure development process management, providing guidance to map organization and application contexts, such as regulatory laws, business environment, technologies, security policies and applications specification. This information allows the definition of security requirements that will result in security controls to mitigate any potential application risk. Then, a customized secure development process is generated after the integration of the identified security controls and a general life cycle model that is employed by the organization as a reference for all project's process definition (ISO/IEC 2011).

Frequently, process tailoring is an important activity performed in an informal and reactive mode, being more expensive, unrepeatable and susceptible to errors (ALEGRÍA; BASTARRICA, 2012). The secure development process results from the security engineering activities integration, that is possible due to process tailoring of several process elements in terms of security requirements, design, coding and test. Then, an inappropriate process tailoring and management can affect the production of secure applications due to the difficulties in manage and control several development processes and their variability (CHATTERJEE; GUPTA; DE, 2013). Another difficulty of the secure development process tailoring is the necessary expertise in application security that demands formal knowledge and experience sharing to assure the security requirements understanding and implementation (WIN et al., 2009).

However, the concept of Software Process Lines (SPrL) can be an alternative to support, organize, manage and control the several number of extended secure development processes that demand an efficient process tailoring approach.

The SPrL or family of processes consists in a set of processes built from a series of shared process assets in order to reuse process knowledge across projects with

different needs. The main benefit is the effort reduction to perform the organization's process management (ARMBRUST et al., 2009).

1.1 Motivation

Secure software development is not a largely explored area for the most part of the organizations that do not integrate the necessary security requirements at early stages of the development process (AGRAWAL; KHAN, 2009). Security aspects must be considered during the whole software development process and the security requirements should be identified, analyzed, designed, coded and tested (ALOTAIBI; LIU, 2014).

Current practices provide guidance on secure software development process definition. However, there is a lack of methods and approaches to support the integration of security activities in the current organizations software life cycle (UZUNOV; FERNANDEZ; FALKNER, 2012). Then resulting in a set of similar processes that must be well defined, documented and managed. The effort to manage all derived processes can be reduced by the use of a specific SPrL for secure development (ARMBRUST et al., 2009).

1.2 Objectives

This work aims to **define a Software Process Line for the secure software development**. A specific SPrL can be useful for organizations that need to manage several secure development processes to conduct different software development projects, considering the process variability that occurs due to the organization and application contexts and the suitable potential risks.

To achieve the proposed general objective, the following specific objectives will be performed:

(i) Identify the most important secure development processes and select their basic elements such as: stages, activities, artifacts, roles and responsibilities.

(ii) Define the Software Process Line based on the variability management of the secure development processes' elements.

(iii) Evaluate the proposed approach.

The main contribution of this work will be a single SPrL model that allows the secure development process, considering the possible process variability that can be

applied to support the organization's process management. Then, the research question that this work intends to answer is "**How the Software Process Line concept can support the secure development process management?**"

1.3 Scope Delimitation

The proposed model will solve the process management problem when organizations integrate security engineering concepts into their software development processes.

1.4 Work structure

To support this research work organization and further execution, it was defined an initial set of phases, activities and expected results. The phases are arranged as following:

- Phase 1 – Research Preparation: phase corresponding to the delimitation of the study area, collection and analysis of bibliographic references, theme delimitation and objectives setting, issues and propositions.
- Phase 2 – Research Structuring: preparation of a theoretical reference framework. Selection of the research method and its stages.
- Phase 3 – Research Execution: stage of the investigation itself, with search and analytical work in the literature, developing the proposed SPrL model that will be applied in a banking industry environment to obtain the necessary results analysis.
- Phase 4 – Results Analysis: stage of the analysis of the data in aggregate form, drawing generalizations and conclusions.

1.5 Dissertation document structure

This document is organized as following:

- Chapter 1 aims to provide to the reader an overview of this research context. Define the main objective and the specific objectives, then presenting the work process.
- Chapter 2 probes the initial theoretical scenario described in Chapter 1, focusing on information security, software development process, security

engineering, secure software development and the standard for security maturity evaluation.

- Chapter 3 presents a methodological position, setting the detailed structure of the research, with the initial proposals based on bibliographic research carried out in Chapter 2.
- Chapter 4 presents the proposed Software Process Line for the secure development.
- Chapter 5 presents the results analysis and discussion, after applying the approach described in the Chapter 4.
- Chapter 6 concludes this work, highlighting the relevance of this work and presenting the final considerations, proposing other future works and researches.

1.6 Chapter considerations

In this chapter, it was possible to understand the importance of the secure software development process to avoid attackers exploiting applications vulnerabilities. Several standards and frameworks provide guidance to the integration of security aspects in an existing software development process. However, there is a lack in terms of process management to support organizations with several secure development processes.

CHAPTER 2 - LITERATURE REVIEW

Those who do not know history are destined to repeat it.
- Edmund Burke

In this chapter, it will be discussed the main concepts of software development process and information security. Then, the approach to assemble them to develop secure software applications.

2.1 Software Development Process

The main objective of software development is to generate products, with high levels of productivity and efficiency that ensure good levels of quality. To achieve this, it is necessary to use different strategies, processes, components and several types of technologies or methods (CASTRO; CRESPO; GARCÍA, 2013).

In the software development area, there are sequential activities that produces a variety of documents and results in a desirable software product. A software development process is a sequence of stages with feedback that enable the software production and further evolution (PETERS; PEDRYCZ, 2001).

A reasonable software development process takes place in an integrated environment that manages the process of product development as well as its evolution. This is possible only if the development process provides the necessary feedback relating to its behavior to the process management and the process management is able to use this information to control the evolution of the process (AJILA; KABA, 2008).

The standard ISO/IEC 12207 (Software life cycle processes) establishes a common framework for software life cycle processes, with well-defined terminology, that can be referenced by the software industry. It contains processes, activities, and tasks that are to be applied during the acquisition of a software product or service and during the supply, development, operation, maintenance and disposal of software products. The standard also provides a process that can be employed for defining, controlling, and improving software life cycle processes (ISO/IEC, 2008a).

Software development life cycles, from traditional models like Waterfall, to more modern ones like RUP, Scrum and XP, suggest specific activities that need to be carried out as part of the development process, as well as the order of these activities. Moreover, if a company wants to certify or evaluate its software development process, this process must be rigorously defined as prescribed by the most popular models and standards (HURTADO et al., 2013).

To asses a software development process, it is possible to use the ISO/IEC 15504 (Process assessment) standard that provides a framework for process assessment. This framework can be used by organizations involved in planning, managing, monitoring, controlling and improving the acquisition, supply, development, operation, evolution and support of products and services (ISO/IEC, 2011).

2.2 Software Product Line

A software product line (SPL) is described by (CLEMENTS; NORTHROP, 2002) as:

> "A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy specific needs of a particular market or mission, and that are developed from a common set of core assets in a prescribed way".

The adoption of SPL can contribute with significant quality and productivity improvement (NEVES et al., 2015).

Core assets are reusable artifacts such as requirements and design documents, software components, project schedules, budgets, test cases, work plans, and process descriptions (CLEMENTS; NORTHROP, 2002). Developing a core asset base requires the following major activities (BACHMANN; CLEMENTS, 2005):

a) Determining which core assets can remain unique and useful for all software products. If necessary, identify the necessary adjustments on them to fit with the set of product's needs (variability).
b) Selecting a variability management mechanism to assure that all required changes keep the core assets widely shared.
c) Providing awareness about how product developers must use the variation mechanisms to include the core assets during the software development process.

The SPL engineering allows the variability management in any organization domain or industry, and it is composed by two distinct processes: domain engineering and application engineering (POHL; METZGER, 2006).

The domain engineering process (shown in Figure 2-1) defines the commonality and the variability of the product line to proceed with the core assets development. The application engineering process is responsible for assemble the available reusable core assets with the application's specific needs. (POHL; METZGER, 2006).

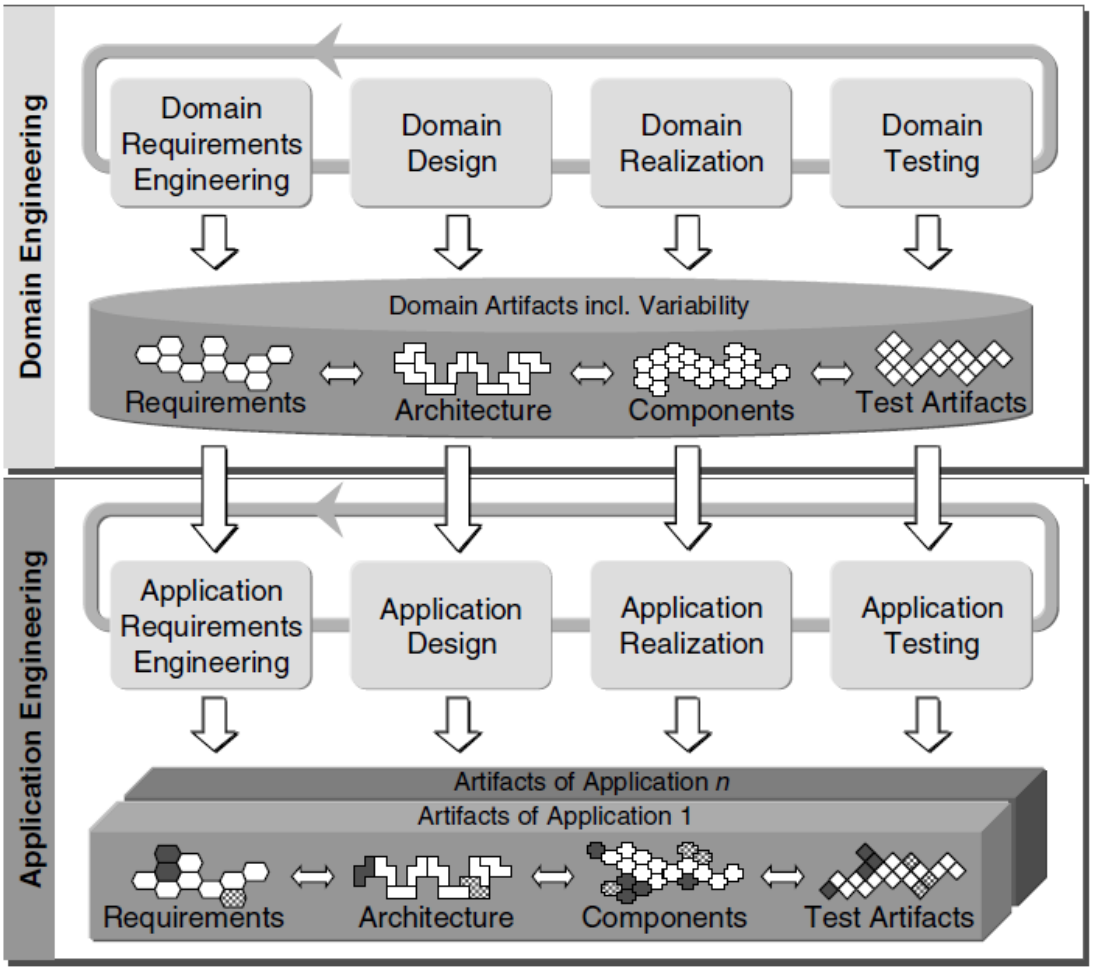

Figure 2-1. Software product line engineering framework (POHL; METZGER, 2006).

Core asset developers define and implement the core assets that will be available to product developers for their use in producing products. Product line managers coordinate and facilitate the work of these two groups as illustrated in Figure 2-2 (MCGREGOR, 2004).

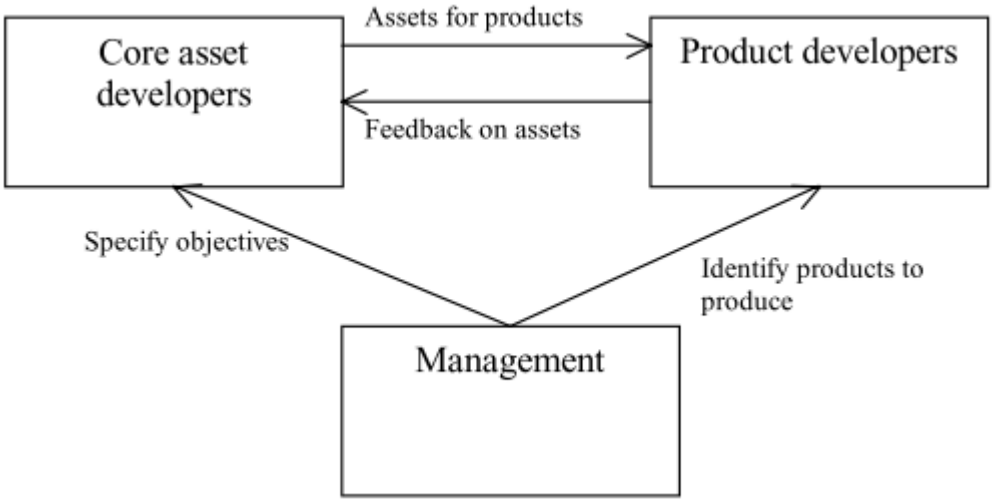

Figure 2-2. Roles in a software product line (MCGREGOR, 2004).

Variability Management encompasses the activities of eliciting and representing variability in software artefacts, establishing and managing dependencies among different variability, and supporting the exploitation of the variability for building and evolving a family of software systems (CHEN; BABAR, 2011).

When the variability management is not well performed, unnecessary variability can be added to the core assets. As the product line grows and evolves, the need for variability increases, and managing the variability grows increasingly difficult (BACHMANN; CLEMENTS, 2005).

In terms of variability modelling, there are two proposed approaches to model the product line variability. Firstly, the integration of the variability notation in existing models using specific stereotypes. As alternative to the integration, another well stablished approach is the use of dedicated structures to manage and control de SPL variability (POHL; METZGER, 2006). The Orthogonal Variability Model (OVM) shown in Figure 2-3 is a dedicated variability model.

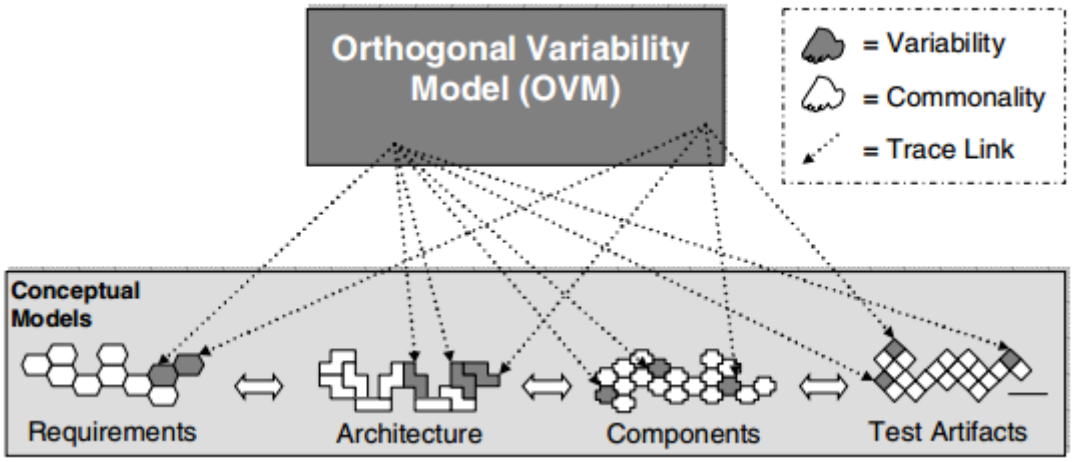

Figure 2-3. Orthogonal Variability Model (POHL; METZGER, 2006).

2.3 Software Process Line

Organizations specify their software development processes to be instantiated several times, expecting reuse knowledge and experiences across the projects development. However, in many cases the project context or environment requires process adjustments and adaptations. Such flexibility is difficult to be managed and controlled (HURTADO et al., 2013).

A Software Process Line (SPrL) is a set of similar processes that shared common process assets such as stages, activities, artifacts, roles and responsibilities. The processes are prepared to support further controlled points of variability due to the projects context or environment (ARMBRUST et al., 2009).

As previously described, the SPL relies on reuse of software core assets to build a derivate software application product with its particularities, producing product families with similar features. Indeed, SPrL specifies a standard process model that can be derivate to define processes for specific project contexts, composing process families with similar process elements (ROUILLE et al., 2013).

The SPrL main benefits are the increase of quality for the generated processes and adherence to the organization context, manage the possible points of variability and common core processes features, reduce the risk of inadequate process customization (LORENZ; BRASIL; FONTOURA, 2014).

Process elements that must be always used in a project are known as mandatory features. The features that can be dismissed or their use is not mandatory are known as optional features. There are process elements that excludes or not allow

the use of other process elements, they are known as alternative features (TERNITE, 2009). The optional and alternative features allow the occurrence of points of variability which type classification is presented in Table 2-1.

Table 2-1. Variability types for process lines (TERNITE, 2009).

Variability Type	Meaning
Positive	New process element addition that not request the exclusion of other process elements.
Negative	Process elements or relations are removed.
Extending	Process elements or relations are extended.
Replacing	Process elements or relations are replaced.

The mechanism to control the process variations is the Software Process Line Architecture (SPrLA). The architecture must allow the configuration of mandatory, optional and alternative features that imply the effects defined by the variability types positive, negative, extending and replacing (TERNITE, 2009). Much research has been published about the SPrLA. The most important contributions are the CASPER (Context Adaptable Software Process EngineeRing) approach and the SPEM (Software and Systems Process Engineering Metamodel Specification) language extension (SCHRAMM; DOHRMANN; KUHTMANN, 2015).

The CASPER is a SPrLA contribution from (ALEGRÍA; BASTARRICA, 2012) that offers a planned software process approach based on four well stablished principles:

- Principle 1 – Separation of Software Process Engineering and Software Engineering domains: Process engineering is focused in defining the overall process line model, considering its variants and adaptation mechanism. The project teams perform the software engineering, when applying the adaptation mechanisms to define their variant processes.
- Principle 2 – Software Process Scoping: process scoping consists in determine when the SPrL can be employed and which process elements (common and variable) will be required in each distinct scenario.
- Principle 3 – Software Process Models are also Software Models: process models can be designed, reused, adapted as made with the software models.

- Principle 4 – Software Process Adaptation Complexity Hiding: the process adaptation do not must require excessive effort from the project team. In this case, the software process engineering defines an adaptable software process model, an adaptation context and a set of tailoring rules to simplify the process tailoring.

The CASPER principles relationship are presented in Figure 2-4.

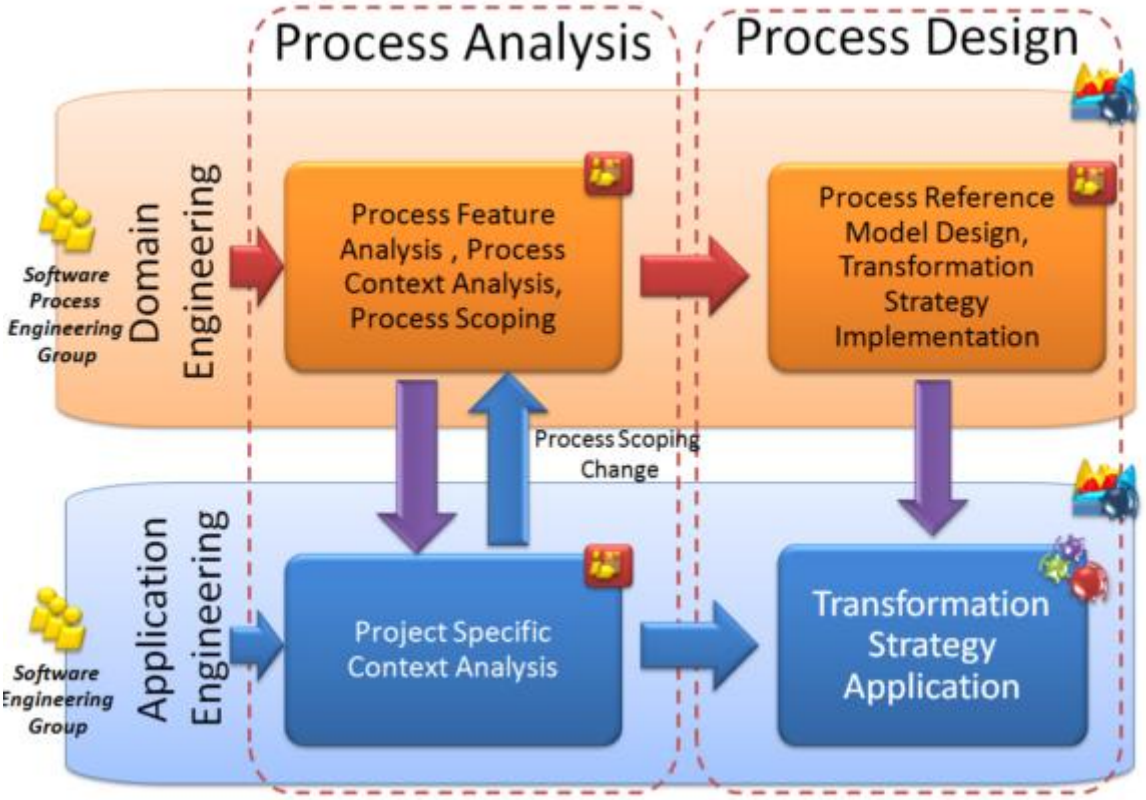

Figure 2-4. SPrL Architecture (ALEGRÍA; BASTARRICA, 2012).

To adopt the CASPER approach, the organization can define two different work teams: one for domain software process engineering (Software Process Engineering Group - SPEG) and another for software process application engineering (Project Team - PT). The PT members will work to define the customized process regarding theirs projects contexts. The SPEG will be in charge of developing and evolving the SPrL (ALEGRÍA; BASTARRICA, 2012).

The SPEM specification is a meta-model for process specification provided by OMG (Object Management Group) based on UML (Unified Modeling Language) specification. SPEM supports the specification of process families and mechanisms to

control the process variation points. This meta-model has a set of stereotypes to represent all process elements and their related variability information (SPEM, 2008).

In terms of SPrL, the combined use of CASPER and SPEM allows the SPrLA definition, specification and management.

2.4 Information Security

Our world is increasingly relying on complex software and systems. In a growing number of fields such as transportation, finance, telecommunications, medical devices, they now play a critical role and require high assurance: any failure caused by an attacker could lead to catastrophic loss in terms of cost, damage to the environment, or even human life (PONSARD et al., 2007).

Information is an asset that, like other important business assets, is essential to an organization's business and consequently needs to be protected. This is especially important in the increasingly interconnected business environment. As a result of this increasing interconnectivity, information is now exposed to a growing number and a wider variety of threats and vulnerabilities (ISO/IEC, 2005).

Information Security (IS) goals are traditionally classified into confidentiality, integrity and availability concepts that compose the CIA triad (ISO/IEC, 2004):

- Confidentiality is the property that information is not made available or disclosed to unauthorized individuals, entities, or processes;
- Integrity is the property of safeguarding the accuracy and completeness of assets;
- Availability is the property of being accessible and usable upon demand by an authorized entity.

This CIA triad is sometimes extended by concepts such as accountability, non-repudiation, and authentication. For example, accountability and non-repudiation can be classified as integrity goals, authentication as a design mechanism to achieve confidentiality or integrity, and anonymity or non-observability may be subsumed under confidentiality goals. Independently of their taxonomy, security goals are defined as very general statements about the security of an asset (FABIAN, 2010).

2.4.1 Security Definitions

Some concepts such as vulnerability, threat, risk, and exposure often are used to represent the same thing even though they have different meanings and relationships to each other. It is important to understand these concepts definition, but more important is to understand its relationship to the other concepts (HARRIS, 2008):

- Vulnerability is a software, hardware, or procedural weakness that may provide an attacker the open door he is looking for to enter a computer or network and have unauthorized access to resources within the environment.
- Threat is any potential danger to information or systems. The threat is when someone, or something, identify a specific vulnerability and use it against the company or individual. A threat agent could be an intruder accessing the network through a port on the firewall, a process accessing data in a way that violates the security policy, a tornado wiping out a facility, or an employee making an unintentional mistake that could expose confidential information or destroy a file's integrity.
- Risk is the likelihood of a threat agent taking advantage of a vulnerability and the corresponding business impact. Risk ties the vulnerability, threat, and likelihood of exploitation to the resulting business impact.
- Exposure is an instance of being exposed to losses from a threat agent. A vulnerability exposes an organization to possible damages.
- Countermeasure, or safeguard, is something put into place to mitigate the potential risk. A countermeasure may be a software configuration, a hardware device, or a procedure that eliminates a vulnerability or reduces the likelihood of a threat agent being able to exploit a vulnerability.

The Figure 2-5 shows these described concepts relationship.

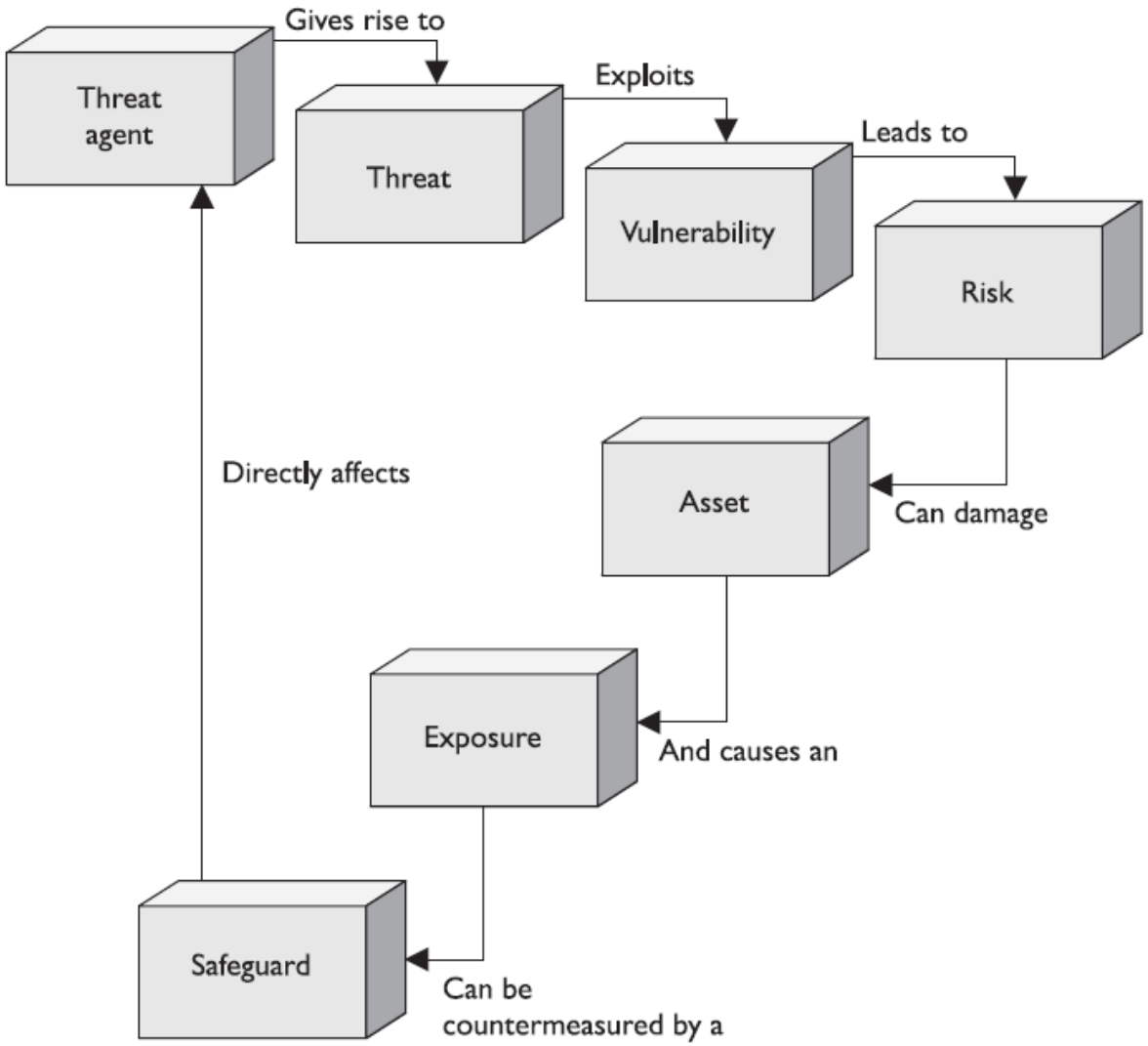

Figure 2-5. Security concepts relationship (HARRIS, 2008).

2.4.2 Risk Management

The NIST (National Institute of Standard and Technologies) that is part of the U.S Department of Commerce, issued a special publication (NIST SP 800-39) that defines risk management as a complex, multifaceted activity that requires the involvement of all organizational levels. Starting with senior managers providing the strategic vision and top-level goals and objectives for the organization; to mid-level leaders planning, executing, and managing projects; to individuals on the front lines operating the information systems supporting the organization's missions/business functions. Risk management is a comprehensive process that requires organizations to (NIST, 2011):

(i) Frame risk: stablishing the risk management strategy that addresses how organizations intend to assess, respond and monitor their risks.

(ii) Assess risk: performing the risk assessment.

(iii) Respond to risk: component of risk management that addresses how organizations respond to risk, after the risk score definition is based on the results obtained from the risk assessments.

(iv) Monitor risk: on ongoing basis using effective organizational communications and a feedback loop for continuous improvement in the risk-related activities of organizations.

The Figure 2-6 illustrates the risk management process and the information and communications flows among components.

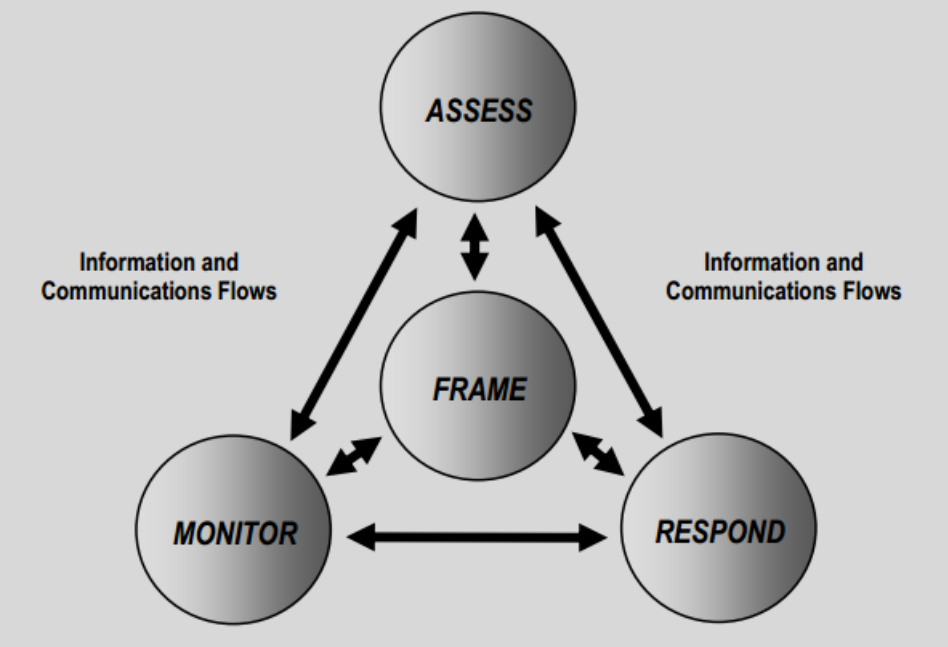

Figure 2-6. Risk management process (NIST, 2011).

The bidirectional nature of the arrows indicates that the information and communication flows among the risk management components as well as the execution order of the components, may be flexible and respond to the dynamic nature of the risk management process. For example, new legislation, directives, or policies may require that organizations implement additional risk response measures immediately. This information has directly communication with the risk-framing component to the risk response component where specific activities are carried out to achieve compliance with the new legislation, directives, or policies, illustrating the very

dynamic and flexible nature of information as it moves through the risk management process (NIST, 2011).

Another NIST special publication (NIST SP 800-37) provides a guide to apply and adopt the risk management concepts in an information system. This guide is composed by 6 steps, as presented in Figure 2-7 (NIST, 2010).

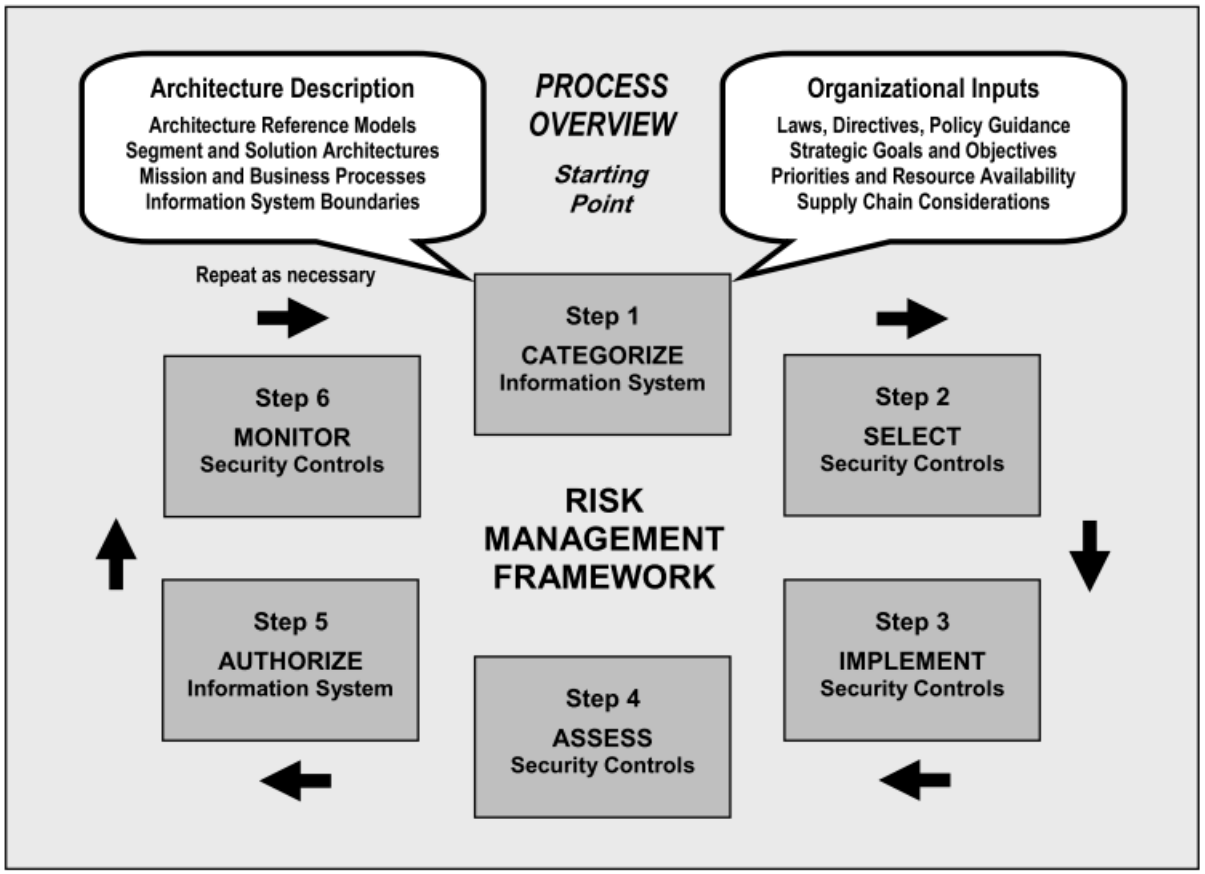

Figure 2-7. NIST definition of Risk Management (NIST, 2010).

The NIST SP 800-37 special publication describes its six steps as following (NIST, 2010):

- Step 1 - Categorize: the information that is inputted, stored and processed must be categorized in terms of its impact analysis.
- Step 2 - Select: select the necessary security controls to avoid any potential risk considering the categorized information.
- Step 3 - Implement: implement the necessary security controls considering the information system operation environment.
- Step 4 - Assess: perform the security control implementation assessment and evaluation to check if they are correctly implemented.

- Step 5 - Authorize: the organization senior manager must accept the resulting risk to authorize the information system operation.
- Step 6 - Monitor: the monitoring step is an ongoing activity to assure the risk management maintenance, assessing any change in the information system.

As described in both NIST special publications, the risk management process requires the employment of risk assessment procedures to complement the activities of risk identification and analysis.

2.4.3 Risk Assessment

Risk assessment is a complex process of identifying, estimating, and prioritizing the mitigation information security risks. Assessing risk requires a detailed analysis of potential threats and vulnerabilities that can cause impact the organization information system (NIST, 2012).

The NIST special publication (NIST SP 800-30) is specific guide for conducting risk assessments. This publication identifies four steps of the information risk assessment process, starting with a review of the existing or proposed system and ending with a commitment to monitor the system on an ongoing basis (NIST, 2012):

- Step 1 – Prepare for assessment: identify the risk assessment context, scope and boundaries, using as reference the frame risk step in the risk management process.
- Step 2 – Conduct assessment: elaborate a list of security risks that must be prioritized by the risk level of each identified risk. The risk level results from the risk impact and likelihood (probability of risk occurrence) combination that can be assisted by a risk matrix.
- Step 3 – Communicate results: communicate and share information about the risk findings to the organization decision-makers.
- Step 4 – Maintain assessment: support the ongoing risk assessment maintenance, reviewing any change in the information system.

The Figure 2-8 presents the relationship of the four described risk assessment steps.

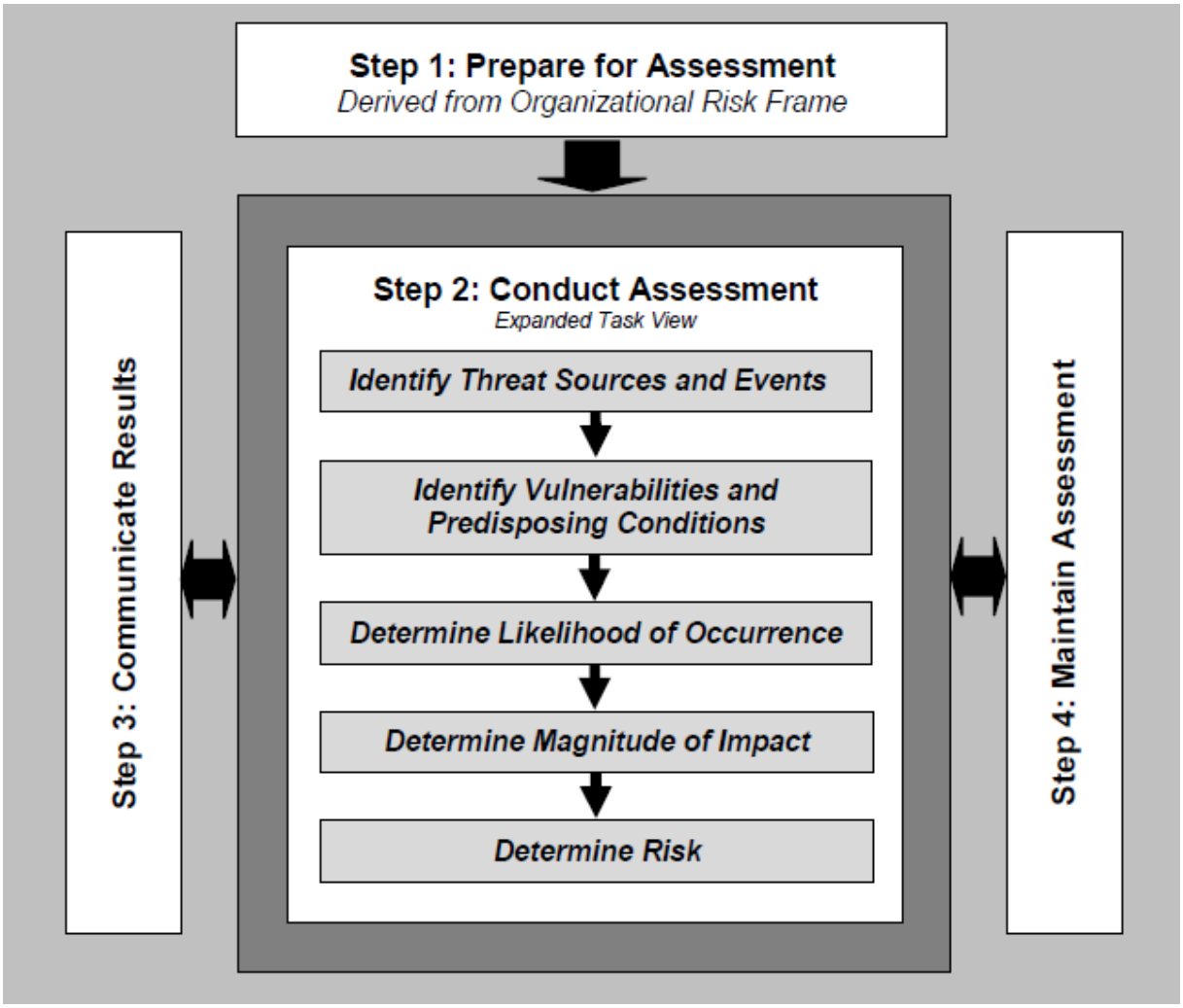

Figure 2-8. Risk assessment process (NIST, 2012).

As mentioned in the Step 2, the risk level can be determined using a risk matrix as presented in Figure 2-9.

		Likelihood		
		Remote	Occasional	Frequent
Impact	Minor	Low	Low	Medium
	Moderate	Low	Medium	High
	Catastrophic	Medium	High	High

Figure 2-9. Risk level matrix (STEWART, 2009).

The risk communication must be efficient and clear to assure that the necessary decisions about the risk mitigation will be provided by the senior managers. Delay in communicating the identified risks or an unclear risk analysis report impacts the risk management process (STEWART, 2009).

The research (MELO, 2008) proposes a framework to inform about vulnerabilities in real-time. This framework combines risk management and risk assessment approaches with a tool that report to the organization decision makers all potential risks in the information systems that required immediate attention.

2.4.4 ISO/IEC 27001 - Information Security Management Standard

The NIST special publications previously presented in this work are guidelines for risk management and risk assessment. However, the standard ISO/IEC 27001 (ISO/IEC, 2013) provides requirements for establishing, implementing, maintaining and continually improving an Information Security Management System (ISMS).

The ISO/IEC 27001 provides a model for implementing the NIST guidelines principles to risk management, risk assessment and a set of security controls (ISO/IEC, 2013). The standard has emphasis on:

- Understanding an organization’s information security requirements and the need to establish policy and objectives for information security.
- Implementing and operating controls to manage an organization’s information security risks in the context of the organization’s overall business risks.
- Monitoring and reviewing the performance and effectiveness of the ISMS.
- Continual improvement based on objective measurement.

This standard adopts the "Plan-Do-Check-Act" (PDCA) improvement approach, which is applied to structure all ISMS processes. Figure 2-10 illustrates how an ISMS takes as input the information security requirements and expectations of the interested parties and through the necessary actions and processes produce information security outcomes that meets those requirements and expectations (ISO/IEC, 2013).

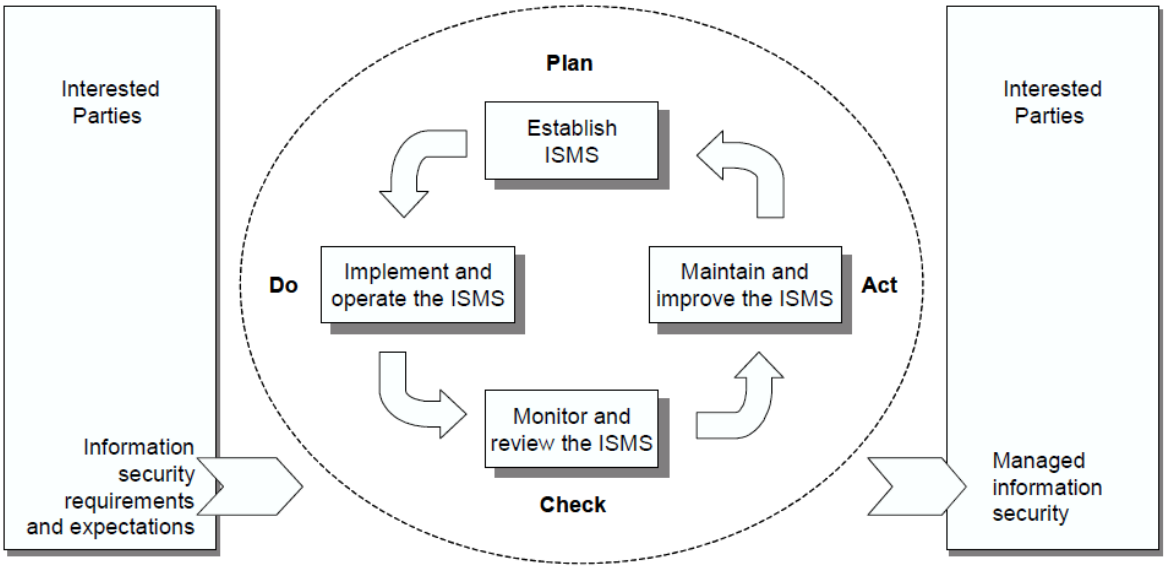

Figure 2-10. PDCA model applied to ISMS processes (ISO/IEC, 2013).

These model stages can be described as (ISO/IEC, 2013):

- Plan: Establish the ISMS policy, objectives, processes and procedures relevant to manage risk and improve the information security system to deliver results in accordance with an organization's objectives.
- Do: Implement and operate the ISMS policy, controls, processes and procedures;
- Check: Assess and measure the process performance and report the obtained results.
- Act: Take corrective and preventive actions, based on the results of the internal ISMS audit and management review to achieve continual improvement of the ISMS.

Management must frequently review the organization's ISMS to ensure its continuing suitability, adequacy and effectiveness. This review can result in opportunities for the ISMS improvement (ISO/IEC, 2013).

The ISO (International Organization for Standardization) organization has other related security information standards that are part of the 27000 family. These are the main standards in the complete ISMS implementation and management:

- ISO/IEC 27000 – Overview: provides the overview of information security management systems (ISMS), and terms and definitions commonly used in the ISMS family of standards.

- ISO/IEC 27002 – Code of practice for information security controls: gives guidelines for organizational information security standards and information security management practices including the selection, implementation and management of controls taking into consideration the organization's information security risk environments.
- ISO/IEC 27003 – Information security management system implementation guidance: focuses on the critical aspects needed for successful design and implementation of an Information Security Management System (ISMS).
- ISO/IEC 27004 – Information security management (measure): provides guidance on the development and use of measures and measurement in order to assess the effectiveness of an implemented information security.
- ISO/IEC 27005 – Information security risk management: it was designed to assist the satisfactory implementation of information security based on a risk management approach.

In terms of ISMS, it is possible to affirm that there are sufficient standards and frameworks to successfully implement and manage a ISMS in an organization.

2.5 Secure Software Development Process

To develop a secure software able to operate correctly after malicious user attacks, it is necessary to employ security engineering activities during the development process. The security requirements must be considered in the project scope and the necessary verification and validation points included in the software development process (OTHMANE et al., 2014).

Security is a non-functional requirement that requires interaction with several systems parts to be efficiently implemented. The decision of not considering the security requirements in their initial development stages, results in an extra effort to integrate the necessary security controls (POPP et al., 2003).

In this section, the approaches to develop secure software will be described. Firstly, the Common Criteria standard that introduces security requirements in IT products. Then, the SSE-CMM model that allows organizations to evaluate their secure development process. Next, the main secure development frameworks that are largely

used and tested by the industry. Finally, it is presented the standard ISO/IEC 27034, that provides guidance to the secure development process definition.

2.5.1 ISO/IEC 15408 – Common Criteria

The security standard ISO/IEC 15408 (Common Criteria for Information Technology Security Evaluation) provides a common set of requirements for the security functionality of IT products and for assurance measures applied to these IT products during a security evaluation (ISO/IEC, 2009). This standard has three parts:

- Part 1: Introduction and general model.
- Part 2: Security functional components.
- Part 3: Security assurance components and Evaluation Methodology.

In this case, the parts 1–3 provide general guidelines to the developers and the customers of IT products, as well as to the evaluators. Note that the Common Criteria (CC) operates under the security assurance paradigm. Security assurance refers to the level of confidence in that the system delivers the specified security functionality, rather than the level of security functionality that often simply referred to as security level (HOUMB et al., 2010).

The main contribution of the CC is a framework that permits comparability between results of independent security evaluations. It is possible by providing a common set of requirements for the security functionality of IT products, and for the assurance measures that are applied to these products during an evaluation. The evaluation process is used to establish confidence in the fulfilment of particular security functionalities (HOUMB et al., 2010).

Therefore, in the context of evaluation, the standard uses the term TOE (Target of Evaluation). While there are cases where a TOE consists of an IT product, this need not be the case. The TOE may be an IT product, a part of an IT product, a set of IT products, a unique technology that may never be made into a product, or a combination of these. Examples of TOEs include (ISO/IEC, 2009):

- A software application.
- An operating system.
- A software application in combination with an operating system.

- A software application in combination with an operating system and a workstation.
- An operating system in combination with a workstation.
- A smart card integrated circuit.
- The cryptographic co-processor of a smart card integrated circuit.
- A Local Area Network including all terminals, servers, network equipment and software.
- A database application excluding the remote client software normally associated with that database application.

The standard ISO/IEC 15408 can be combined with the standard ISO/IEC 12207 that establishes a common framework for software life cycle processes with well-defined terminology, that can be referenced by the software industry (ISO/IEC, 2008a). The resulted combination is a set of secure development processes.

2.5.2 Microsoft SDL

The Microsoft Security Development Lifecycle (SDL) is a secure development process that focuses on software development. The SDL embeds security best practices within the software development process. Since 2004, Microsoft has been employing SDL internally to develop software products (REAVIS, 2013).

SDL comprises a set of activities, which complement Microsoft's development process and which are particularly aimed at addressing security issues. SDL can be characterized as follows (WIN et al., 2009):

- Security as a quality attribute: the primary goal of SDL is to increase the quality of functionality-driven software by improving its security posture.
- Process definition: the SDL process is organized in stages that are co-related with the software development stages.
- Guidance: provide a set of guidelines to integrate security activities in the software development process.
- Management: offer a management perspective for the elicitation and description of the security activities.

2.5.3 OWASP CLASP

The Open Web Application Security Project (OWASP) is a worldwide not-for-profit charitable organization focused on improving the security of software. This organization defined a secure software development process named CLASP (Comprehensive, Lightweight Application Security Process) that provides a well-organized and structured approach for moving security concerns into the early stages of the software development lifecycle, whenever possible (OWASP, 2014).

CLASP is actually a set of process pieces that can be integrated into any software development process. It is designed to be both easy to adopt and effective. It takes a prescriptive approach by documenting activities that organizations should perform and provides an extensive wealth of security resources that make implementing those activities reasonable (OWASP, 2014). The CLASP components are:

- Institute awareness programs: all organization staff involved with the software development must be trained in essential security concepts and techniques.
- Application assessment: the risk assessment on the organization applications must be performed to assure the software quality.
- Security requirements: the security requirements must be identified and elicit to assure the secure development process definition.
- Secure development practices: the secure development process and its elements must be defined and documented.
- Remediation procedures: define which steps will be taken to identify, assess, prioritize and remediate vulnerabilities.
- Metrics: define and monitor metrics to assess the current organization security posture, focusing attention on the most critical vulnerabilities, and reveal how well the investments in improved security are performing.
- Security guidelines: provide stakeholder with documentation on operational security measures and functions that can better secure the product.

To be effective, best practices of software application security must have a reliable process to guide a development team in creating and deploying a software application that is as resistant as possible to security vulnerabilities (OWASP, 2014).

2.5.4 McGraw Touchpoints

The Touchpoints framework provides a set of best practices that have been distilled over the years out of the extensive industrial experience of its proposer. Most of the best practices are grouped together in seven so-called touch points. Touchpoints can be characterized as follows (WIN et al., 2009):

- Risk Management: Touchpoints acknowledges the importance of risk management when it comes to software security. It tries to bridge the gap by elaborating a Risk Management Framework (RMF).
- Black vs. White: The touch points provide a mix of black-hat and white-hat activities, both of which are necessary to come to effective results. Black-hat activities are about attacks, exploits and breaking software (e.g., penetration testing). White-hat activities are more constructive in nature and cover design, controls and functionality (e.g., code review).
- Flexibility: the touch points can be tailored to the software development process already in use. To facilitate this, the documentation provides a prioritization of the different touch points. This allows companies to gradually introduce the touch points, starting from the most important ones.
- Resources: provides links to resources and also explains how to use them. For instance, attack patterns are provided in order to be used in the elicitation of abuse cases.

The Touchpoints is rich on examples. For instance, when describing abuse cases, there is an example giving the reader a good feel about what they might look like in a particular situation (WIN et al., 2009).

2.5.5 ISO/IEC 21827 – SSE - CMM

The maturity of any secure development processes can be evaluated using the standard ISO/IEC 21827 that specifies the Systems Security Engineering - Capability Maturity Model® (SSE-CMM®), which describes the essential characteristics of an organization's security engineering process that must exist to ensure good security engineering. ISO/IEC 21827 (ISO/IEC, 2008b) does not prescribe a particular process

or sequence, but captures practices generally observed in industry and can be used as a

- Tool for engineering organizations to evaluate their security engineering practices and define improvements.
- Method by which security engineering evaluation organizations such as certifiers and evaluators can establish confidence in the organizational capability as one input to system or product security assurance.
- Standard mechanism for customers to evaluate a provider's security engineering capability.

This International Standard has a relationship to ISO/IEC 15504 (Information technology – Process assessment), particularly ISO/IEC 15504-2 (Performing an assessment), as both are concerned with process improvement and capability maturity assessment. However, the standard ISO/IEC 15504 is specifically focused on software processes, whereas the SSE-CMM® is focused on security (ISO/IEC, 2008b).

SSE-CMM® Architecture

The SSE-CMM® architecture is designed to enable a determination of a security engineering organization's process maturity across the breadth of security engineering. The goal of the architecture is to clearly separate basic characteristics of the security engineering process (domain dimension) from its management and institutionalization (capability dimension) characteristics (ISO/IEC, 2008b).

The SSE-CMM® contains 129 Base Practices (BP), organized into 22 Process Areas (PA), as presented in Table 2-2. Of these, 61 base practices, organized in 11 process areas, cover all major areas of security engineering. The remaining 68 base practices, organized in 11 process areas, address the project and organization domains (ISO/IEC, 2008b).

Table 2-2. SSE-CMM Process Areas (ISO/IEC, 2008b)

SSE-CMM® Process Areas	
Domain – Security Engineering	**Capability – Management Process**
PA01 Administer Security Controls	PA12 - Ensure Quality
PA02 Assess Impact	PA13 - Manage Configuration
PA03 Assess Security Risk	PA14 - Manage Project Risk
PA04 Assess Threat	PA15 - Monitor and Control Technical Effort
PA05 Assess Vulnerability	PA16 - Plan Technical Effort

PA06 Build Assurance Argument	PA17 - Define Organization's Systems Engineering Process
PA07 Coordinate Security	PA18 - Improve Organization's Systems Engineering Process
PA08 Monitor Security Posture	PA19 - Manage Product Line Evolution
PA09 Provide Security Input	PA20 - Manage Systems Engineering Support Environment
PA10 Specify Security Needs	PA21 - Provide Ongoing Skills and Knowledge
PA11 Verify and Validate Security	PA22 - Coordinate with Suppliers

SSE-CMM® Capability Levels

Organizing the practices into capability levels provides to the organization an improvement guidance, allowing to enhance its capability for a specific process. For these reasons, the practices in the SSE-CMM® are grouped into common features, which are ordered by capability levels, as presented in Table 2-3 (ISO/IEC, 2008b).

Table 2-3. SSE-CMM Capabilities Levels (ISO/IEC, 2008b)

SSE-CMM Capability Levels	
Level	**Description**
Level 1 – Performed Informally	The Performed Informally level focuses on whether an organization performs a process that incorporates the base practices.
Level 2 – Planned and Tracked	The Planned and Tracked level focuses on project level definition, planning and performance issues.
Level 3 – Well Defined	The Well Defined level focuses on disciplined tailoring from defined processes at the organization level.
Level 4 – Quantitatively Controlled	The Quantitatively Controlled level focuses on measurements being tied to the business goals of the organization.
Level 5 – Continuously Improving	The Continuously Improving level gains leverage from all the management practice improvements seen in the earlier levels, then emphasizes the cultural shifts that will sustain the gains made.

An assessment should be performed to determine the capability levels for each of the process areas. This indicates that different process areas can and probably will exist at different levels of capability (ISO/IEC, 2008b).

2.5.6 ISO/IEC 27034 – Application Security

A specific standard for secure software development was released in 2011. The ISO/IEC 27034 provides guidance to assist organizations in integrating security into the processes used for managing their applications. In addition, it is applicable to in-house developed applications, applications acquired from third parties, and where the development or the operation of the application is outsourced (ISO/IEC, 2011).

The purpose of ISO/IEC 27034 (ISO/IEC, 2011) is to assist organizations in integrating security seamlessly throughout the life cycle of their applications by:

- Providing concepts, principles, frameworks, components and processes.
- Providing process-oriented mechanisms for establishing security requirements, assessing security risks, assigning a Targeted Level of Trust and selecting corresponding security controls and verification measures.
- Providing guidelines for establishing acceptance criteria to organizations outsourcing the development or operation of applications, and for organizations purchasing from third-party applications.
- Providing process-oriented mechanisms for determining, generating and collecting the evidence needed to demonstrate that their applications can be used securely under a defined environment.
- Supporting the general concepts specified in ISO/IEC 27001 and assisting with the satisfactory implementation of information security based on a risk management approach.
- Providing a framework that helps to implement the security controls specified in ISO/IEC 27002 and other standards.

The requirements and processes specified in ISO/IEC 27034 are not intended to be implemented in isolation but rather integrated into an organization's existing processes. To this effect, organizations should map their existing processes and frameworks to those proposed by ISO/IEC 27034, thus reducing the impact of implementing ISO/IEC 27034. This standard will be composed by five parts:

- Part 1: Overview and concepts.
- Part 2: Organization normative framework.
- Part 3: Application security management process.
- Part 4: Application security validation.
- Part 5: Protocols and application security control data structure.

The standard Part 1 was released in 2011 and there is not a plan to release the other five parts that are in development.

Application Security Management Process

To implement an Application Security Management Process (ASMP) the organization will have to create a committee who will manage this overall application security process. This ASMP committee will ensure the process is the answer to the organization's application security concerns and that it is applied to all application projects in the organization. The Application Security Management Process is composed by five steps as presented in the Figure 2-11. (ISO/IEC, 2011).

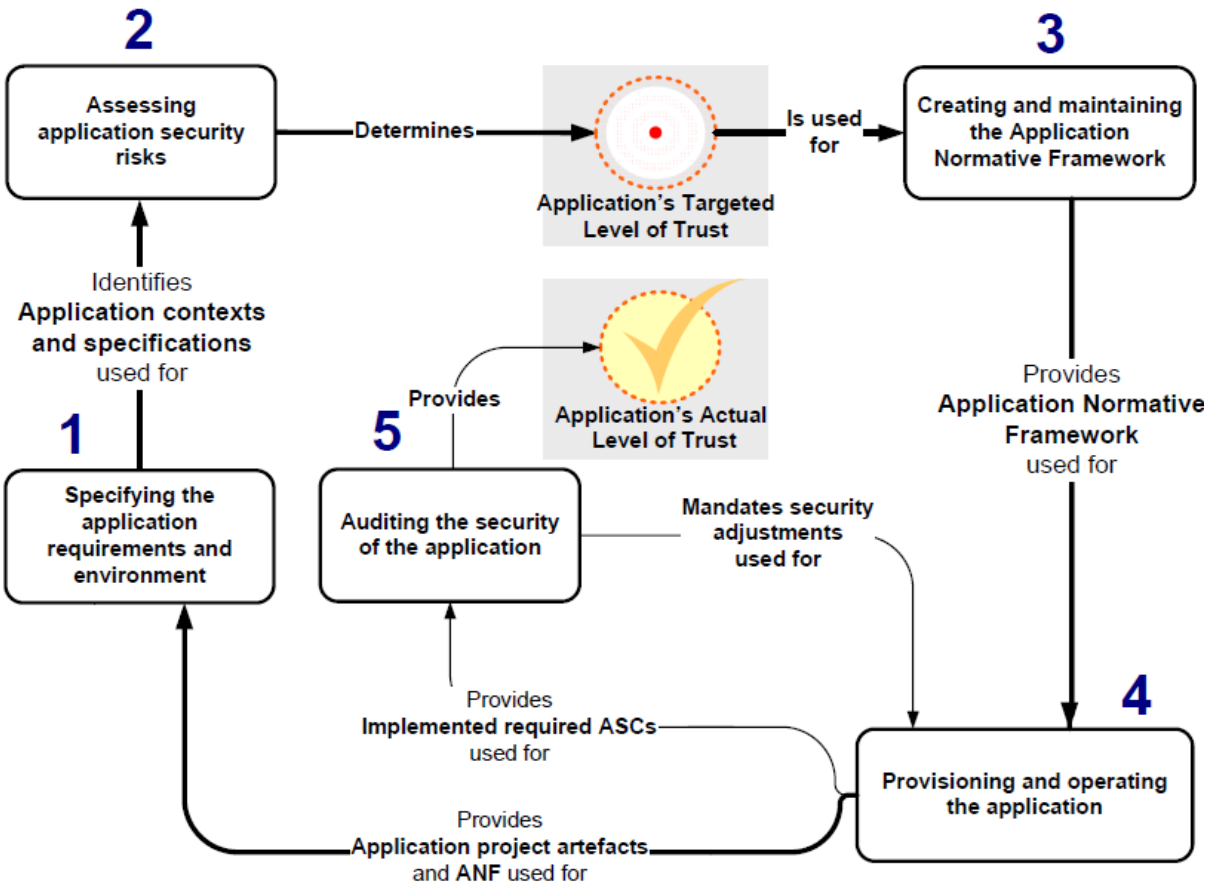

Figure 2-11. Application Security Management Process (ISO/IEC, 2011).

Specifying the application requirements and environment

The first step of the ASMP consist in reviewing the application requirements and its related environment to identify security characteristics that will be used in the security analysis phase (ISO/IEC, 2011).

Assessing application security risks

The second step of the ASMP is a process corresponding to the risk assessment step and a part of the risk treatment step in the risk management process established by ISO/IEC 27005, with a finer granularity level and a scope limited to a single application project (ISO/IEC, 2011).

This step of the ASMP also produces security requirements, which are used to obtain the desired level of trust for the application. This is called as application's Targeted Level of Trust. It should be approved by the application owner (ISO/IEC, 2011).

Creating and maintaining the Application Normative Framework

The third step of the ASMP selects all the relevant elements from the Organization Normative Framework (ONF) that apply to a specific application project. This results in the Application Normative Framework (ANF). The application's Targeted Level of Trust, the application contexts (regulatory, business and technological), the actors' responsibilities and professional qualifications, and the application specifications determine the exact contents of the ANF (ISO/IEC, 2011).

It is also during this step that the organization derives the life cycle for the application project, which contains only those activities needed for the application project. For example, a project developed entirely in-house does not require outsourcing activities. In addition, the organization selects the applicable Application Security Controls for the application project (ISO/IEC, 2011).

Provisioning and Operating the Application

The fourth step of the ASMP is the actual use of the Application Security Controls, as provided by the ANF in the application's life cycle. The project team implements the Application Security Controls (ASC) under the ANF, in two sub-steps (ISO/IEC, 2011):

- The security activity part of each ASC is performed by the corresponding actor assigned in the ASC.
- The security measurement that is part of each ASC, it is performed by the corresponding actor assigned in the ASC.

Auditing the application security

The fifth and final step of the ASMP is the security audit of the application. In this step, a verification team verifies that all the verification measurements provided by all the ASCs in the Application Normative Framework have been performed and that the expected results were attained (ISO/IEC, 2011).

This process may be performed by an internal or an external verification team, using the controls provided by the Application Normative Framework (ISO/IEC, 2011).

The purpose of this step is to verify and provide evidence that an application has reached and maintained the targeted level of trust. It will measure the actual application level of trust at a specific time. Depending of the level of trust needed for the particular application project, this process may be unique, periodic, or event-driven (ISO/IEC, 2011).

Organization Normative Framework

The ONF contains all the regulations, laws, best practices, roles and responsibilities accepted by the organization. It defines all organization contexts and becomes the unique organization referential for application security (ISO/IEC, 2011).

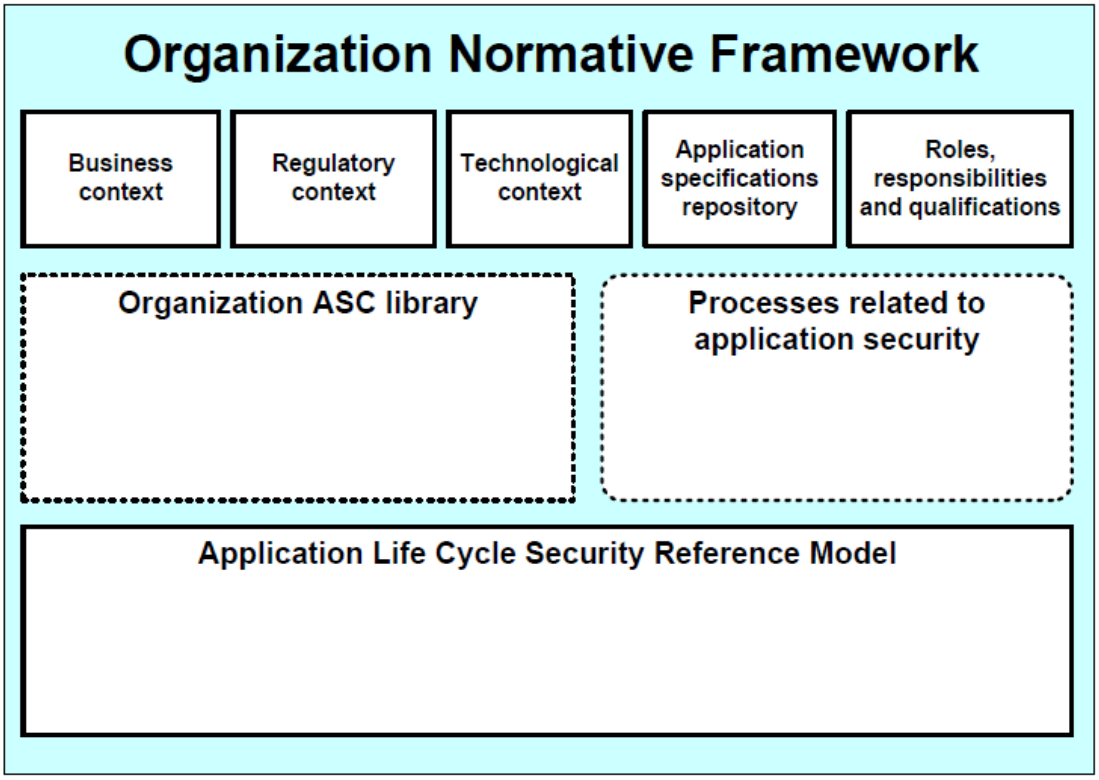

Figure 2-12. Organization Normative Framework (ISO/IEC, 2011).

The ONF is the foundation of application security in the organization and all future application security decisions will be made by referring to this framework. For example, code reviews can only be performed in a project if coding guidelines can be found in the ONF. The ONF components are (ISO/IEC, 2011):

- Business Context: the business context is a list and documentation of all standards and best practices adopted by the organization that may have an impact on business application projects.
- Regulatory Context: The legal context is a list and documentation of all laws and regulations that may have an impact on business application projects, in any of the organization's business locations.
- Technological Context: The technological context is an inventory of all products and technologies available for application projects in the organization.
- Application Specification: list and documentation of the organization's usual functional requirements and corresponding pre-approved secure solutions.
- Roles, Responsibilities and Qualifications: list and documentation of all roles, responsibilities and required qualifications for actors involved in the organization's security application lifecycle.

Figure 2-13 shows how roles and responsibilities are formally specified in terms of ONF and ANF relationship. The verification team is in charge of validate the use of application security in the project and organization levels (ISO/IEC, 2011).

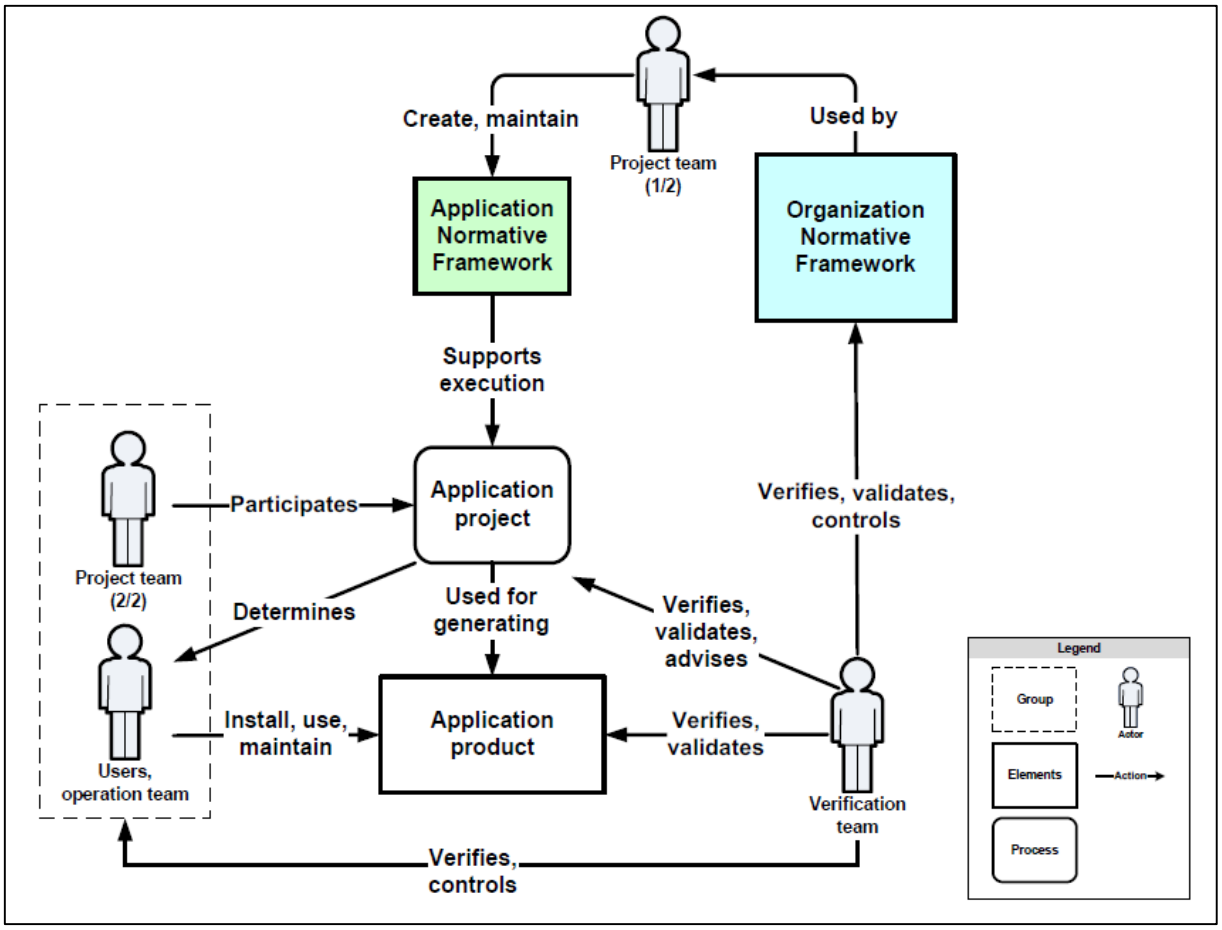

Figure 2-13. Project impact by the use of ISO/IEC 27034 (ISO/IEC, 2011).

The ONF's components are inputs for the ASC Library. This library is a list and documentation of all ASC's used by the organization, attached to the standards, best practices, actors, users, contexts and application characteristics that they evolved from, in relation to the organization's defined levels of trust (ISO/IEC, 2011).

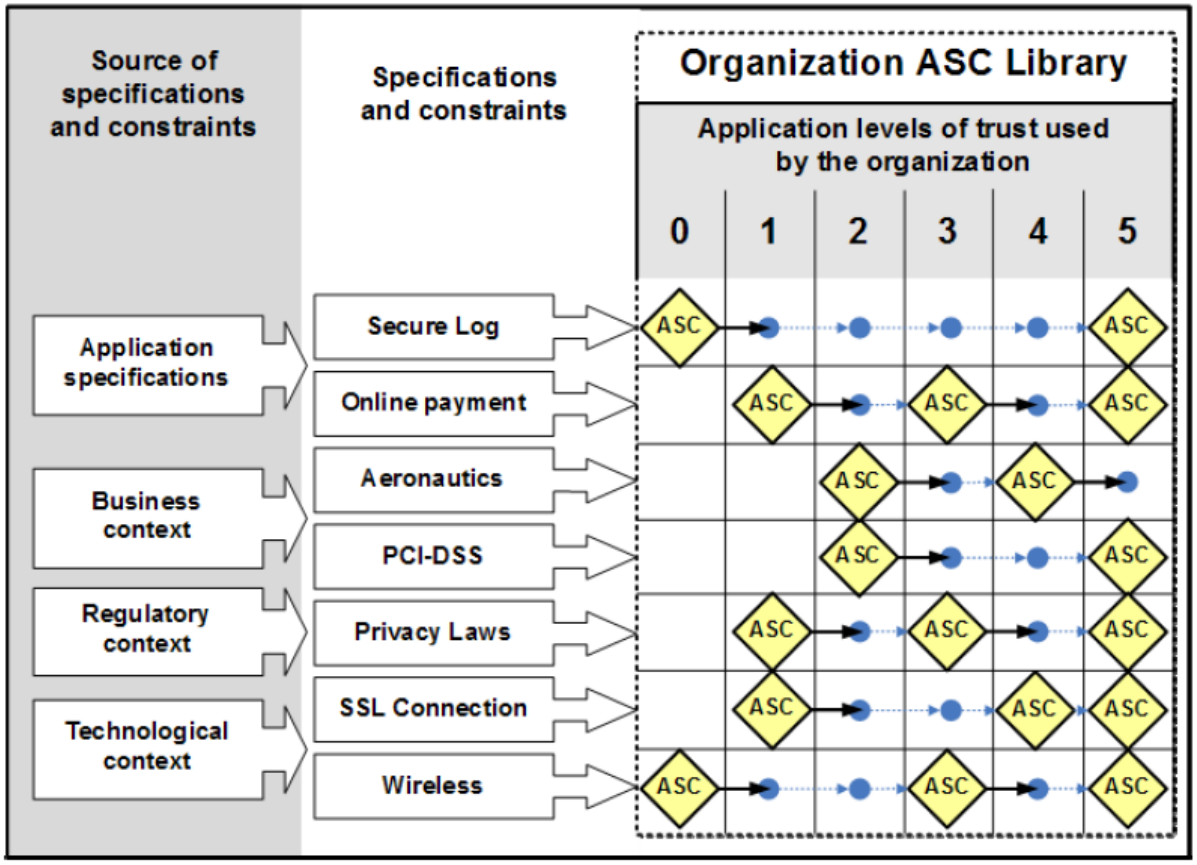

Figure 2-14. Application Security Control Library (ISO/IEC, 2011).

From this library will be selected the ASCs needed for any specific business application project. The Figure 2-14 presents an example of how an organization could use the ASC Library to identify the level of trust of a specific application. The organization must define its own range, or scale, of levels of trust that can be selected as a target for business applications (ISO/IEC, 2011).

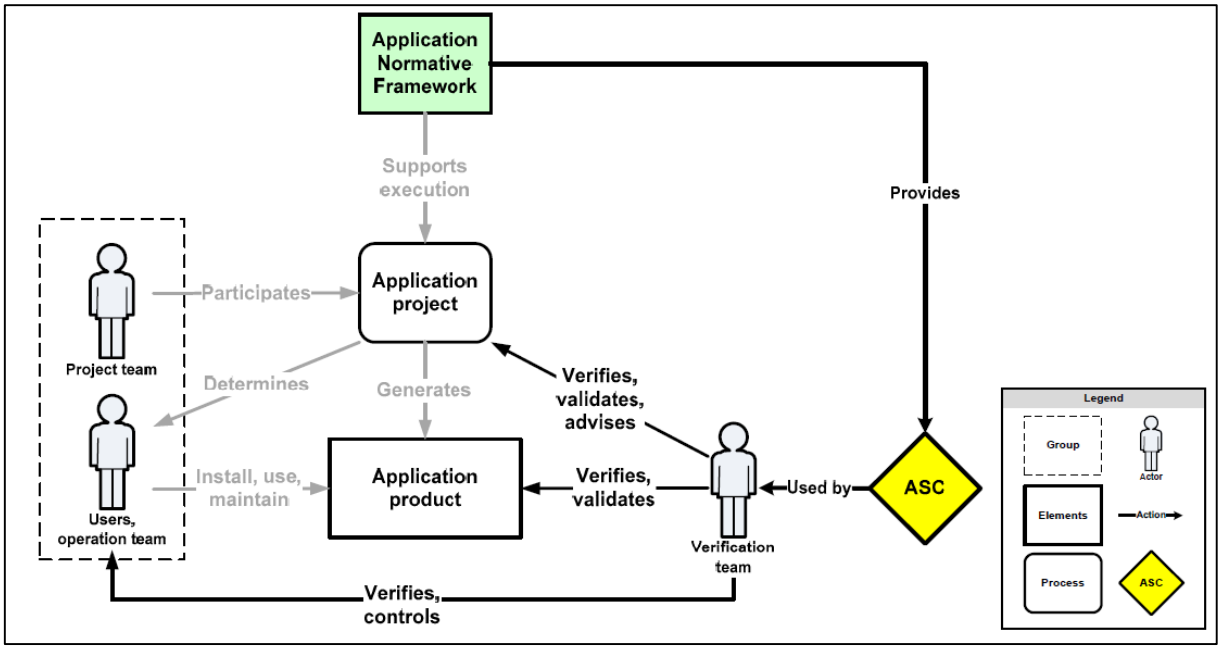

Figure 2-15. Application Security Control (ISO/IEC, 2011).

Figure 2-15 shows the ASC used as a control in a business application project by the verification team (ISO/IEC, 2011).

2.6 Related works

The Evidence-based Software Engineering (EBSE) is a research approach that aims to apply an evidence-based approach to software engineering research and practice. In this context, evidence is defined as a synthesis of best quality scientific studies on a specific topic or research question. The main method of synthesis is a Systematic Literature Review (SLR). In contrast to an expert review using ad hoc literature selection, a SLR is a methodologically rigorous review of research results (KITCHENHAM et al., 2009). In this case, the SLR was planned and performed as following:

1. Research questions definition;
2. Search plan and execution process;
3. Data analysis.

In the sequence, these mentioned stages were carried out.

2.6.1 Research questions definition

In this stage, the set of research questions were defined based on the project objectives. Then, the research questions are:

- RQ1 - How the SPrL concept integration can support the secure development process management?
- RQ2 - Which variability management techniques can be employed?

2.6.2 Searching process

The manual search was planned to answer the research questions. A set of keywords was defined to support the search into the literature repository. The keywords combination used in this search was:

(“software process line” or “software process line engineering”) AND (“secure development” OR “secure software development”)

These keywords combination were used as input in the engine search databases and repositories listed below:

- ACM Digital Library
- Elsevier Campus
- IEEE Explore
- Science Direct
- Scopus

2.6.3 Data analysis

The manual search was performed, but no related works were found using the previous informed search criteria. The lack of findings demonstrates that there is not related works covering the use of SPrL to secure development.

In the literature, secure software development is a subject with different approaches.

Security requirements was initially discussed by (LORIN, 1985). Then, all its main methodologies compared by (FABIAN et al,, 2010). (MELLADO et al., 2010) performed an extensive systematic review which contributed to another studied (MELLADO; MOURATIDIS; FERNANDEZ-MEDINA, 2014) that integrated SPL with secure development. In addition, (Faegri; Hallsteinsen, 2006) applied SPL in software architecture development.

In (BARTSCH, 2011) is demonstrated that secure Agile development is a challenge in terms of life-cycle definition and management. Then, authors focused in use punctual Agile methods such as the Feature-Driven Development by (SIPONEN; BASKERVILLE; KUIVALAINEN, 2005) and Scrum by (MOUGOUEI; SANI; ALMASI; 2013). Next (OTHMANE et al., 2014) suggests the simple addition of security activities in the project interactions without any process derivation or management. The selection of main process elements from Microsoft SDL and CLASP to define an Agile development process was discussed by (BACA; CARLSSON, 2011) without the use of SPrL. All the mentioned Agile approaches, are examples of software process customization for secure development, but no one applied the SPrL concept to achieve all potential process management benefits.

There is a gap of contributions in terms of use SPrL to develop secure software. As previously discussed in this work, the use of SPrL is motivated by the complexity of the secure development processes that are composed by many points of variability in their structures. In this case, the use and extension of secure development process can be improved by the employment of SPrL.

2.7 Chapter considerations

The Information Security is an important field that must be largely explored by all organizations, indifferently of their size, type or industry. In this chapter, the ISMS concepts were described to enable the discussion of security standards and frameworks that can provide the necessary guidance to develop secure products.

CHAPTER 3 - RESEARCH STRUCTURE

This is not the end. It is not even the beginning of the end.
But it is, perhaps, the end of the beginning
- Winston Churchill

This chapter describes the research techniques and approaches to achieve the main work's goal.

3.1. Relevant concepts about research methodology

Research in common parlance refers to a search for knowledge. One can also define research as a scientific and systematic search for pertinent information on a specific topic. In fact, research is an art of scientific investigation (KOTHARI, 2004).

The research process can be classified using the following three categories considering their objectives (GIL, 2002):

- Exploratory research: The objective is to provide greater familiarity with the problem, making it more explicit or to hypotheses building, improving the ideas or discovering insights. It has a flexible planning, considering various aspects of the studied fact. Most of these studies involve: a) Literature; b) Interviews with people that have practical experience with the problem; and, c) Analysis of examples to encourage understanding.
- Descriptive research: The primary objective is the description of the characteristics of a given population or phenomenon or the establishment of relationships between variables. There are countless studies that can be classified in this light and one of its most significant features is the use of standard techniques of data collection, such as the questionnaire and systematic observation.
- Explanatory Research: The objective is to identify the factors that determine or contribute to the occurrence of the problem. Then, it is the most complex and delicate type, since the risk of making mistakes increases considerably. It is possible to assume that scientific knowledge is seated on the results offered by explanatory studies.

However, it is also possible to classify the research methods using their technical procedures. In this case, the classification uses the information source that can be documents or specialized people (GIL, 2002):

- Bibliographic Search: developed based on materials already developed, consisting mainly of reference books and scientific articles.
- Document Search: similar to the bibliographical research, but with materials that have not yet received an analytical treatment.
- Experimental Research: consists of determining an object of study, select the variables that influence it, define the forms of control and observation of the effects of these variables on the object.
- Search Ex-post Fact: (Search from the past event) is a study after occurrence of variation in the dependent variable in the natural course of events.
- Research Study Cut: refers to a group of people who have some characteristic in common, constituting a sample to be accompanied by time period in relation to the fact investigated.
- Survey Research: it is characterized by direct questioning of people whose behavior you want to know.
- Field Study Search: similar to the survey, but in greater depth. Are used more means of observation than question. Typically focuses on a community. The researcher does most of the work in person.
- Case Study Research: refers to the deep and comprehensive study of one or a few objects, allowing a detailed knowledge.
- Action Research: has the active involvement of the researcher and action by individuals or groups involved in the problem.
- Participant Research: it is characterized by the interaction between researchers in which the researcher takes part of the action.

3.2. Research characterization

Based on the research methodology concepts and the work's objectives, it is possible to assume that:

- Considering the research objectives: Exploratory

- Technical procedure implementation: Action Research

The action research is an interactive approach that requires an intensive researcher involvement and it is composed by three types of steps (COUGHLAN; COGHLAN, 2002):

- Pre-step: context and research purpose understating and definition.
- Six main steps: obtaining and processing the context information to stablish an action plan that will be implemented and evaluated.
- Meta-step: monitoring the execution of the six main steps and identify the need for additional cycle planning and execution.

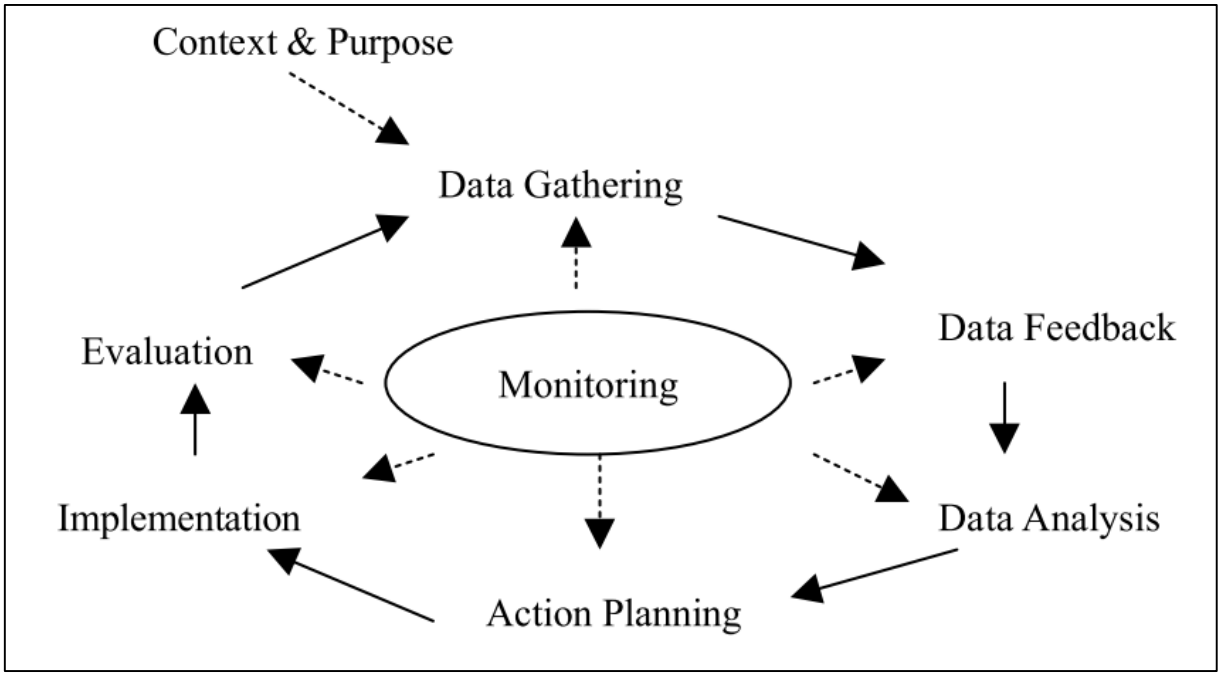

Figure 3-1. The action research types of steps (COUGHLAN; COGHLAN, 2002).

3.3. Research Strategy

This research was performed in three distinct stages to achieve the work's goals and answer the main research question:

- Stage 1: Define the context and stablish the research purpose (pre-step).
- Stage 2: Analyze the organization process development scenario, performing the data gathering, data feedback and data analysis activities.

- Stage 3: Define the proposed SPrL for secure development, identifying the most important secure development processes elements that will be part of the process and application domains.
- Stage 4: Prepare the action plan in accordance with the identified process issues to implement the SPrL.
- Stage 5: Implement the SPrL for secure development based on the variability management of the necessary secure development processes elements.
- Stage 6: Evaluate the proposed SPrL secure process.

After each stage completion, there will be meetings with the involved organization's members to capture and register their feedback. In this case, potential research improvement points will be discussed to provide the necessary clarifications.

3.3.1. Stage 1 – Define the context and stablish the research purpose

The organization is an international banking company in Brazil. This company has offices and operations in 70 countries in all continents, serving some 51 million customers.

The organization offers several banking services such as commercial banking, insurance, investments, currency exchange, global trade, private banking and other international banking services.

To offer all banking services, the organization has a complex system platform composed by several applications that are tailored to be deployed in different countries and regions. In this case, the IT department must address all possible regional regulations and laws (components variability).

All applications must be developed using the security aspects. The secure development process is performed and its outputs are shared with the project teams. At the end of the secure development, all necessary evidences and findings are documented and reported to the senior managers and project teams. When necessary, the reports are shared with regulatory agencies.

The organization project teams perform the process tailoring in an informal way, generating several similar processes which, in the most part of the time, dismiss mandatory process activities due to problems in their definition. As the secure

development processes do not match with the organization development standards, the generated software cannot be attested as reliable.

In this case, the SPrL concept can be applied to improve the process management. The organizational software development process can be flexible for each project development team, being the project team in charge of its project life cycle model definition and process customization. Many points of variability will be necessary to assist the projects process customization management and to assure the secure development process mandatory elements. For this reason, the SPrL concept must be also taken into consideration.

3.3.2. Stage 2 - Analyze the organization process development scenario

In the second stage, the current organization processes will be analyzed to proceed with a formal scenario review, as following:

1. Data gathering: all necessary organization and projects context information will be collected and documented, including process elements, metrics and performance indicators.
2. Data feedback: the obtained information will be reported to the organization stakeholders that are part of the secure development process.
3. Data analysis: the reported data will be analyzed and discussed with the project team (collaborative approach).

The main organization process issues will be discussed with the stakeholders. This stage is critical to guide the SPrL definition that must be elaborated to solve the identified issues.

Before starting with the SPrL introduction, all projects process elements (activities, roles, outputs etc.) will be identified and documented. This is necessary to obtain a complete overview of the current organization processes and scope delimitation.

3.3.3. Stage 3 - Define the proposed SPrL for secure development

The third stage of this work, will consider the process findings provided in the previous stage to offer the necessary solution. It will be performed the selection of process elements from the ISO/IEC standards (27001, 27034, 21827), frameworks (Microsoft SDL, OWASP CLASP and McGraw Touchpoints) and the SPrL engineering

method. The analysis of these process elements results in a set of variability points that will guide their integration and use in the proposed SPrL for secure development.

The stage defines the proposed SPrL for secure development that will use the CASPER principles (ALEGRÍA; BASTARRICA, 2012):

- Separation of Software Process Engineering and Software Engineering domains: all selected process elements (previous stage) will be mapped, documented and their use recommended to the overall process engineering or only for the software application engineering domains. In this case a domain matrix will be elaborated to better represent the separation of domains.
- Software Process Scoping: a SPrL document that will determine when the SPrL can be employed and which process elements (common and variable) will be required in each distinct scenario.
- Software Process Models are also Software Models: the SPEM language will be employed to model the general reference process that will represent the SPrL and its points of variability.
- Software Process Adaptation Complexity Hiding: to adapt the SPrL in several scenarios, a tailoring guide document will be elaborated.

3.3.4. Stage 4 - Prepare the action plan

In the fourth stage, the organization management and the researcher will plan the secure SPrL implementation. The resources and team members will be assigned to accomplish with the SPrL definition and adoption in a proper timeline, considering the plan constraints.

The proposed secure SPrL will be introduced in this organizational environment and applied in a set of projects. As the organization projects are developed in phases (interactive and incremental life cycle approach), the model will be applied and its results will be properly measured, collected and analyzed every 2 or 4 weeks (average project's phase duration).

3.3.5. Stage 5 - Implement the SPrL for secure development

The fifth stage consists in the action plan execution. The plan activities are monitored and controlled by the organization managers and the researcher.

3.3.6. Stage 6 - Evaluate the SPrL implementation

The final stage consists in the proposed SPrL evaluation, considering the information obtained during the process definition and implementation. It will be necessary to verify if the mandatory security activities were successfully integrated and performed by the project teams. It will be possible also, measure the effort to performer mandatory, optional and alternative process elements. The implementation stage outcomes are reviewed, the lessons learnt are registered.

To support the SPrL evaluation, a process assessment will be performed in the organization process before the SPrL introduction and after its implementation. The SSE-CMM standard will be applied as an assessment model to proceed with this analysis.

3.4. Chapter considerations

This chapter presented the relevant concepts to structure this research, establishing an initial classification of the types of research that will be addressed in this work. The steps of the research were detailed in order to clarify and define the procedures to be performed during the execution of this work.

CHAPTER 4 - RESEARCH DEVELOPMENT

This is not the end. It is not even the beginning of the end.
But it is, perhaps, the end of the beginning
- Winston Churchill

This chapter describes the research development and its results after using the action-research method.

4.1. Preliminary phase: context and purpose

The original organization process was created in 2010 by a group of security engineering specialists. The most part of them are working in other departments or left the organization. In this case, no changes or reviews were made since the process definition.

The process starts with a Project Security Review (PSR) request sent by the business area to the application secure development team. This request consists in a set of project information such as project description, list of impacted systems and components, stakeholders and project planning.

Then, the secure development team selects a Security Project Manager (SPM) to be engaged in this review request. The SPM engages a Security Engineer (SE) specialist to review the project scope (project requirements and design documents) and identify potential threats that can be exploited by malicious users. These threats are analyzed to measure their risks. All information generated by the SE specialist will be reported in the Risk Assessment (RA) document. The SE specialist provide a Work Breakdown Structure (WBS) document after receiving the necessary documents and before starting the threat analysis.

The RA document must be presented to its review requester and the development team that will be aware about the project's initial risks.

Next, the Security Tester (that was engaged by the SPM) will plan the security test phase (white and black box testing). This planning consists in the Security Test Plan document that covers the security test cases elaboration, and the Test Report document that will report all security test findings and their evidences.

The business requester and the development team are again informed about the findings and their risk rating. In an iterative lifecycle, the development team can understand the recommended security controls by the Security Engineer and Security Tester specialists to proceed with their development and integration in the next project iteration.

Finally, after all project iterations the SPM produces a Final Business Report document that reports all project risks that were identified during the project development and were not fixed by the project team.

For each not fixed project finding, the business requester must inform a tentative date to implement the necessary solution. In the further project releases, the Security Project Manager will be supposed to recover the previous Final Business Summary in the team repository and provide it to the Security Engineer and the Security Tester. Table 4-1 contains the roles and artifacts relationship for the described secure development process.

Table 4-1. Original process roles and artifacts relationship (Author).

	Business Requester	**Development Team**	**Security Project Manager**	**Security Engineer**	**Security Tester**
Project Security Review Request	Provide		Use		
Requirement Specification	Provide	Provide	Use	Use	Use
Design Solution Specification		Provide		Use	
Security Review Plan	Use	Use	Provide	Use	Use
Security Engineering WBS			Use	Provide	
Risk Assessment		Use	Use	Provide	Use
Security Test Plan		Use	Use		Provide
Security Report		Use	Use		Provide
Final Business Report	Use	Use	Provide	Use	Use
Business Sign-Off	Provide		Use		

4.2. First Improvement Cycle

The first improvement cycle was performed using the Action-Research method phases: data gathering, data analysis, data feedback, action plan, implementation and evaluation.

4.2.1. Data Gathering, Analysis and Feedback

To start this cycle a research group initially composed by five secure development specialists was defined to support the Action-Research on the target organization environment.

The research group was supposed to use the original organization secure development process as reference. This process is based on Microsoft SDL (Microsoft Software Development Lifecycle). However, a successful SPrL should not consider just a single framework or standard. In this case, the proposed SPrL should be generic enough to be employed in several organizational environments.

The proposed SPrL should be documented using any specialized tool for process definition in order to facilitate the reuse by further users and works. To support the proposed SPrL instantiation, a tailoring guide should be elaborated to provide the necessary understanding about the process variation points and when the SPrL could be employed.

Initially, the data gathering, analysis and feedback stages were performed and the research group identified and listed the main goals for the first improvement cycle:

1. Analyze the main secure processes to identify their commonalities and variabilities;
2. Select a specific tool to specify the proposed SPrL;
3. Identify when and how the SPrL can be employed;
4. Elaborate a Tailoring Guide document.

The first cycle goal is to perform the separation of Software Process Engineering and Software Engineering domains and the Software Process Scoping to identify the possible variability points, as suggested by the CASPER approach.

4.2.2. Action planning

Analyzing the requirement of not defining the SPrL to be compatible just with a single secure development process, it was necessary to consider the main method contents offered by the most important secure development processes.

In WIN et al.(2009) the three main secure development processes (CLASP, Microsoft SDL and McGraw) were evaluated and compared in detail. This work described the commonalities among these secure development processes and also their specificities. It was possible to expand the compared secure processes analysis to identify the overloaded activities and their phases that are part of the proposed secure SPrL, as presented in Table 4-2.

Table 4-2. Resulting analysis from processes commonalities and variabilities (Author).

Management & Training	Requirements	Design
- Core Security Training (M) - Institute Security Awareness Program (C) - Monitor Security Metrics (C) - Address reported security issues (C,M)	- Establish and Document Security Requirements (M, C, T) - Create Quality Gates/Bug Bars (M) - Perform Security and Privacy Security Assessments (M) - Specify Operational Environment (C) - Identify Global Security Policy (C) - Detail Misuse Cases (C,T) - Perform Risk Analysis (T)	- Perform Attack Surface Analysis/Reduction (M,C,T) - Use Threat Modeling (M,C) - Identify User Roles and Resources Capabilities (C) - Apply Security Principles to Design (C,M) - Research and Assess Security Posture of Technology Solutions (C) - Specify Database Security Configuration (C) - Perform Risk Analysis (T)
Implementation	**Verification**	**Release**
- Use Approved Tools (M) - Deprecate Unsafe Functions (M) - Perform Static Analysis (M,T) - Integrate security analysis into source management process (C) - Implement interface contracts (C)	- Perform Dynamic Analysis (M) - Perform Fuzz Testing (M) - Perform source-level security review (C) - Identify, implement, and perform security tests (M,C,T) - Verify security attributes of resources (C) - Perform Risk Analysis (T)	- Create an Incident Response Plan (M) - Conduct Final Security Review (M,C) - Perform code signing (C) - Build operational security guide (C,T)

(M) Microsoft SDL, (C) CLASP, (T) Touchpoints

As requested in the previous stage, the SPrL process must be specified and documented. In this work, vSPEM language was pointed by the literature review chapter as the appropriate language to represent the suggested secure process as this language provide a formal representation to the variation points.

Some works such as GenArch-P (ALEIXO et al., 2011) and SOPLA (GARCIA, 2012) provided some tools that are based on the Eclipse Process Composer (EPF). However, these tools are not available anymore by their respective authors due to technical problems[4].

The SmartySPEM (JUNIOR et al., 2013) is an UML based profile that allows the process variability representation using specific stereotypes (see Table 4-3). This profile can be used with the Enterprise Architect modeling tool. However, this tool does not provide process definition and documentation support as the EPF tool, in terms of method contents, artifacts, roles etc.

Table 4-3. SmatySPEM stereotypes (JUNIOR et al., 2011).

Stereotype	Description
	Mandatory task that must be always performed. The user do not have the option of dismiss the task.
	Optional task that can be included or not in the project process derived from the SPrL.
	Condition that offers one or more options of tasks that can be selected and performed. The user would have 2 or more tasks options and can select 1 or all of them.

[4] Information obtained by e-mail.

	Exclusion of tasks options, in this case the just one task can be selected to the process. This condition is used when having 2 or more options and the user must select only one of them.

As alternative, the research group team employed the SmartySPEM to represent the SPrL and its variabilities and then to produce the Tailoring Guide document. The SPrL process documentation is performed by the EPF tool that does not support vSPEM, but will be manipulated by the SPrL users with the Tailoring Guide document assistance.

4.2.3. Implement the SPrL for secure development

A secure SPrL was created taking into account phases and activities previously presented in Table 4-2. The workflow in Figure 4-1 presents the possible phases transitions along a project development using the secure SPrL.

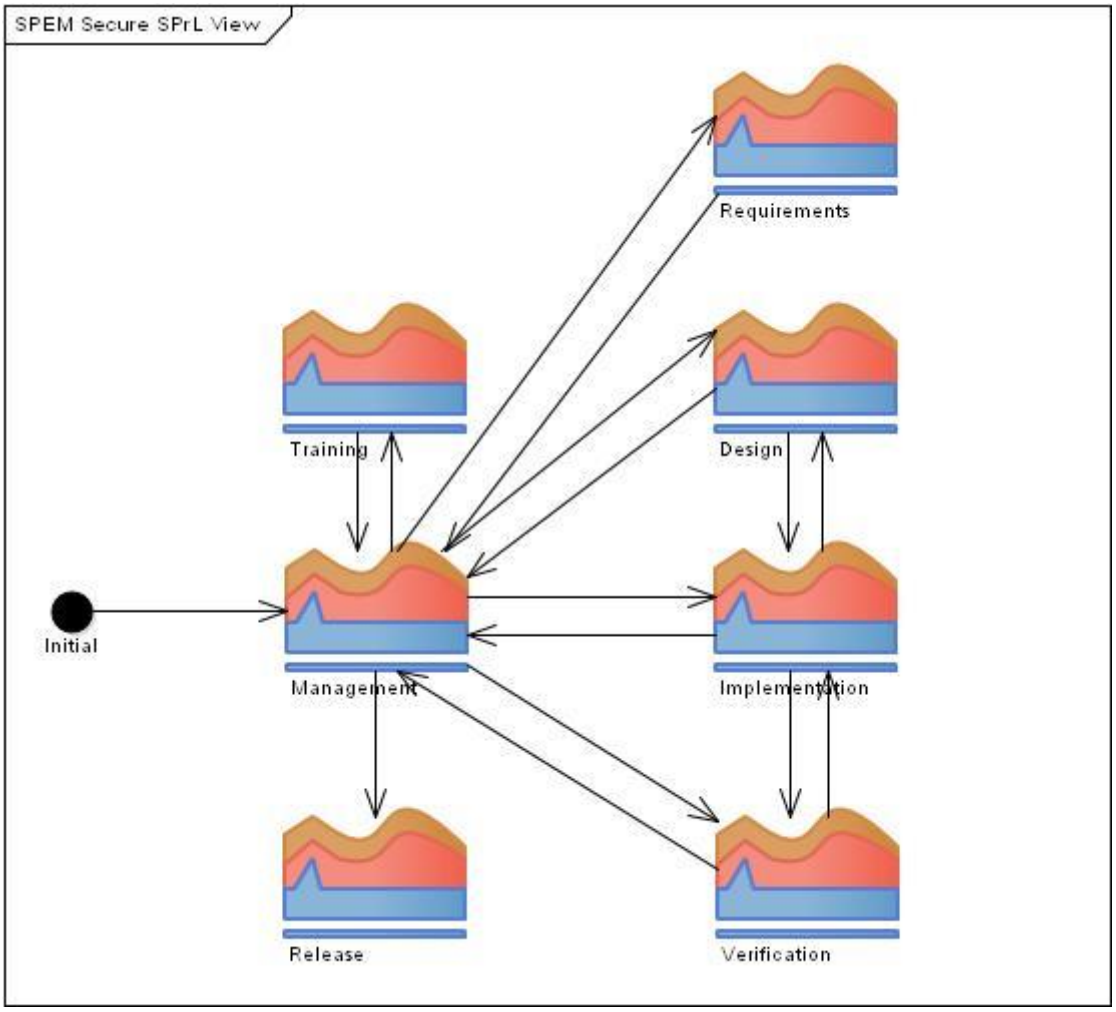

Figure 4-1. SPrL phases workflow (Author).

The main phase is Management because this phase centralizes the effort of monitoring the security metrics that can be raised in any process activity and must be addressed and reported to the organization's senior management board, as can be seen in Figure 4-2. Both phases are mandatory and can be performed in parallel.

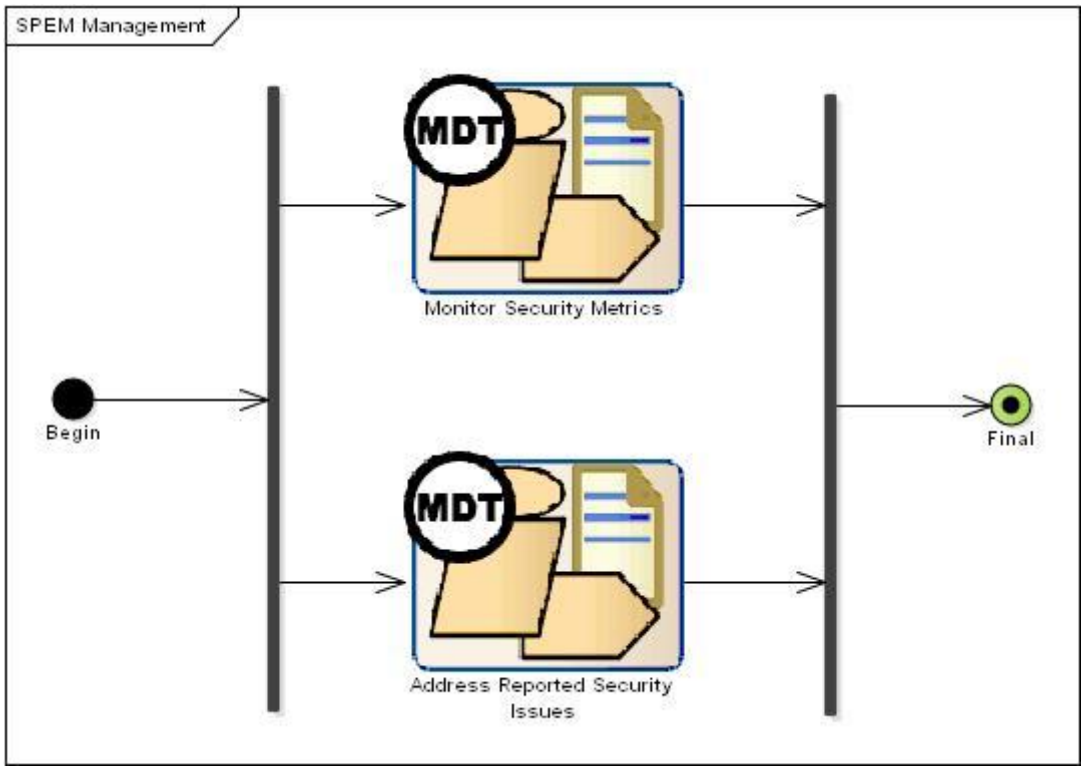

Figure 4-2. Management phase overview (Author).

The Management phase also starts the Training phase that provides and assure the necessary security knowledge to the project team and the minimum-security awareness to the overall organization's staff (optionally) as presented in Figure 4-3.

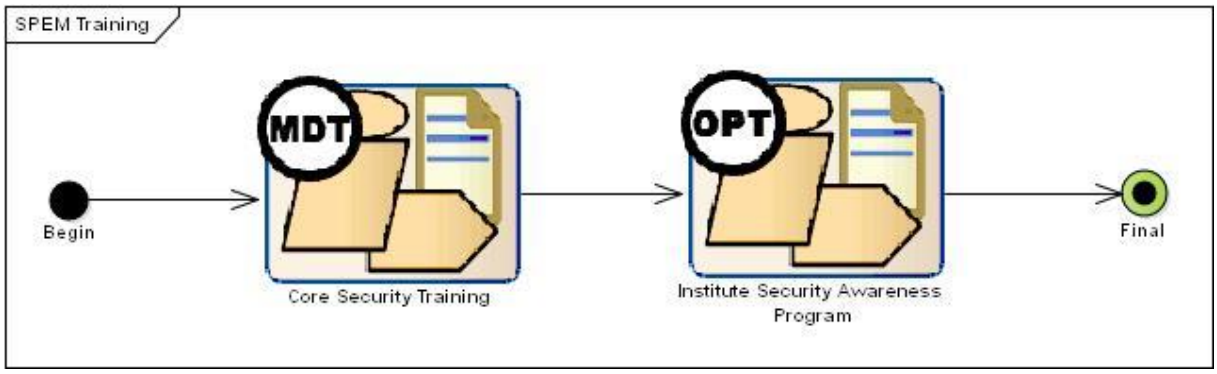

Figure 4-3. Training phase overview (Author).

The Requirements phase (see Figure 4-4) consists in the security requirements analysis and documentation. The solution requirements related to systems features are also specified and documented in this phase. In this case the first phase's activity named as Document Security Requirements covers all solutions requirements, being in this case mandatory.

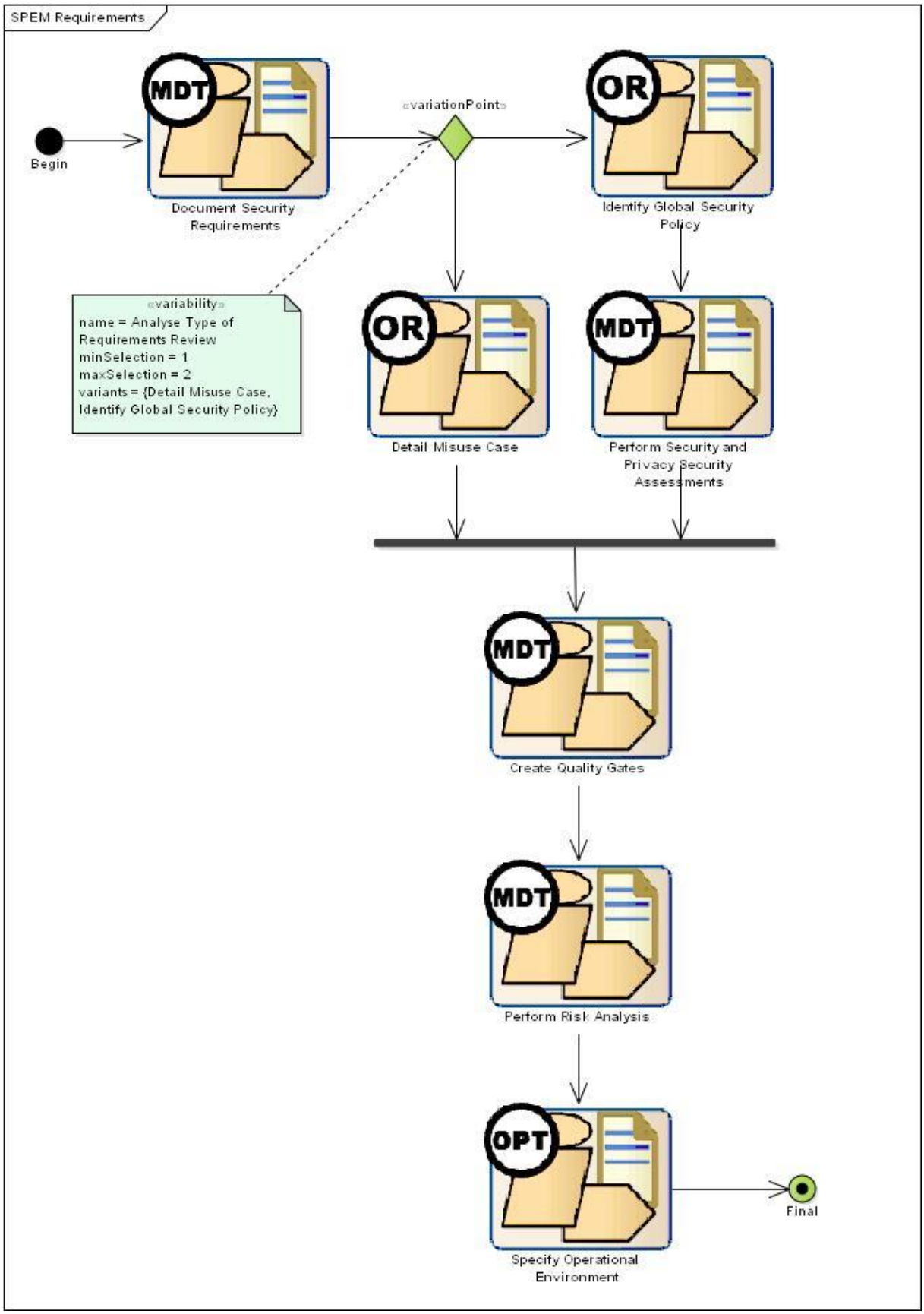

Figure 4-4. Requirements phase overview (Author).

After the Document Security Requirements activity, the phase has a variation point that reflects the project type of security requirements review approach. There are two available options: Detail Misuse Case and Identify Global Security Policy activity. The first one can be selected in case of using Use Case modeling analysis approach. The second one is recommended when working focused in identifying potential risks to the organization's security policy caused by the solution's requested features. Next, the Create Quality Gates activity is performed to support the creation of quality bars that can be an outcome to the Management phase (Monitor Security Metrics activity).

The Perform Risk Analysis activity is mandatory, because all potential risk identified during the Requirements phase must be evaluated and documented to further mitigation. To complete this phase, the optional Specify Operational Environment activity can be performed to assure a complete solution environment overview.

The Design phase (see Figure 4-5) starts with the Research and Assess Security Posture of Technology Solutions activity that covers potential risks in the technologies (third party components, frameworks, protocols etc) in use by the project. Then, the Specify Database Security Configuration activity must be executed to assure security of all data stored and manipulated by the solution.

The Design phase has a variation point, because there are two possible approaches to perform the security design review. The first approach begins with the Establish Design Requirements activity, that covers a solution design based on security aspects introduced in the design artifacts and models, especially in the class diagrams. The second one, is based on the Threat Modelling activity that is preceded by the Identify User Roles and Resources Capabilities activity and the Identify Resources and Trust Boundaries activity.

Resuming the Design phase, it is necessary to execute the Perform Risk Analysis activity, to assure that potential risks identified in this phase were correctly reported to further mitigation.

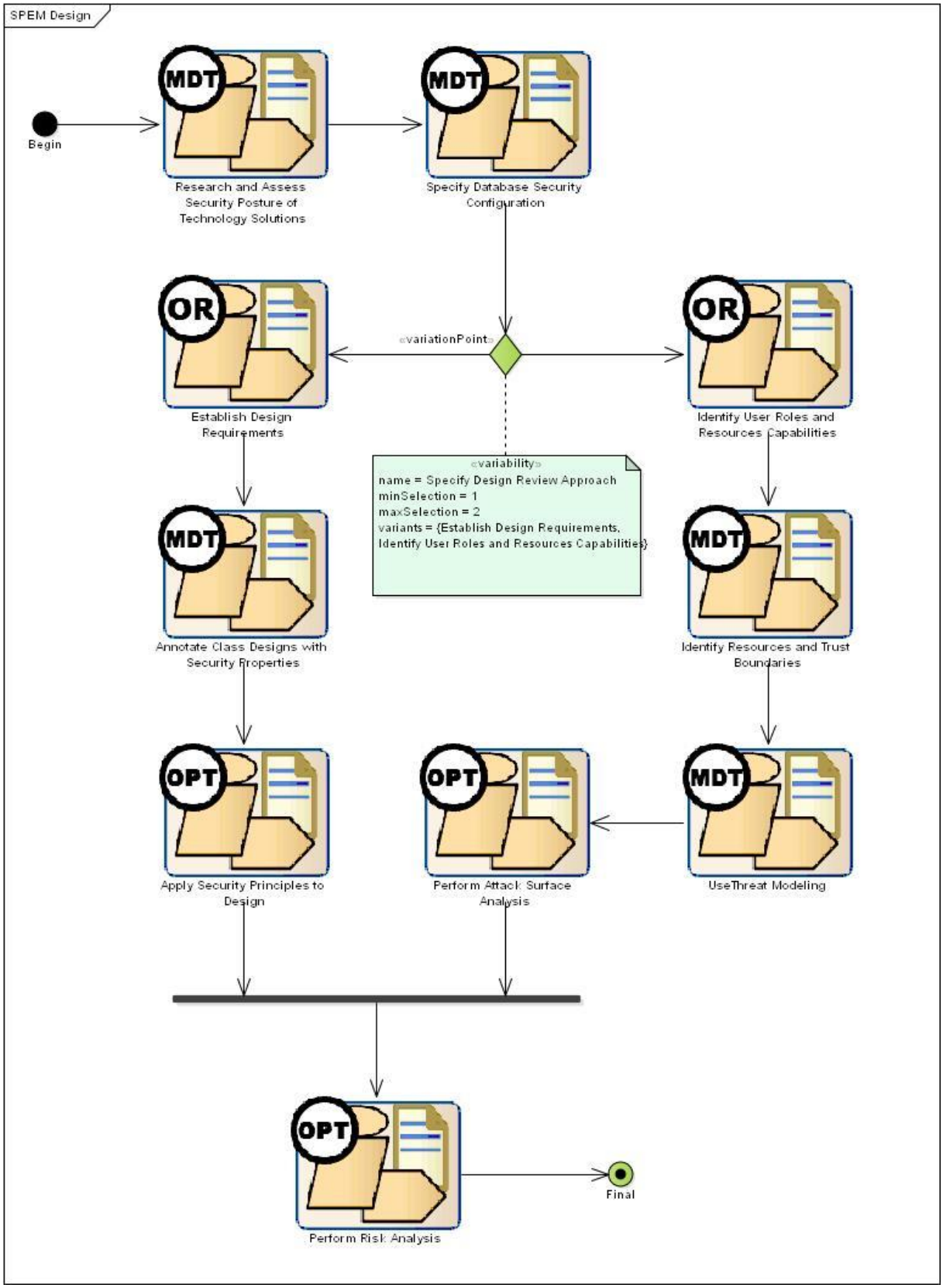

Figure 4-5. Design phase overview (Author).

The Implementation phase (see Figure 4-6), starts with the Elaborate and Implement Resource Policies and Security Technologies activity that is followed by two optional activities: Deprecate Unsafe Functions and Implement Interface Contracts.

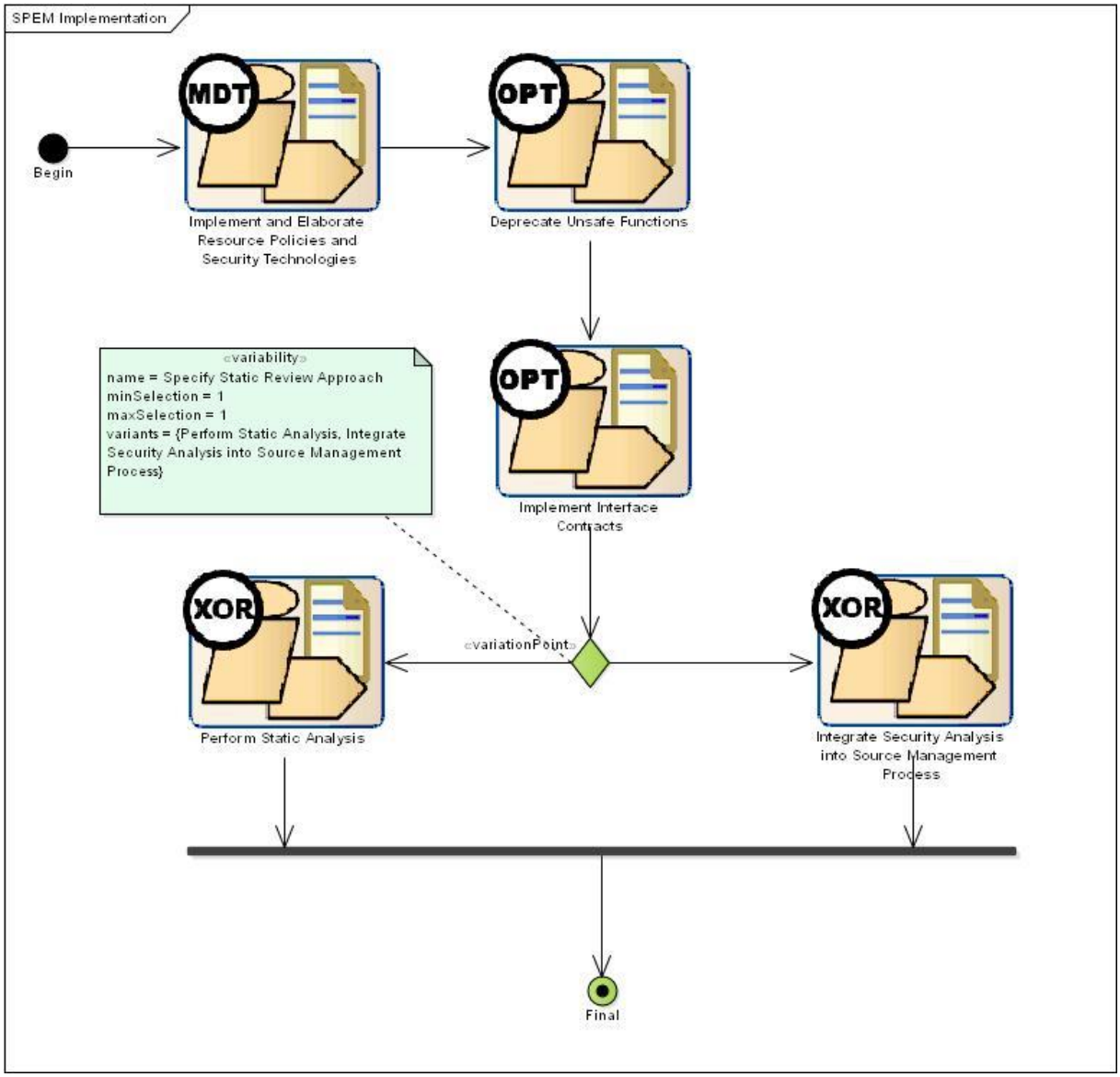

Figure 4-6. Implementation phase overview (Author).

There is a variation point in the Implementation phase, the source-code can be reviewed through the Perform Static Analysis activity or automatically when some code is added in the project repository. In this case, executing the Integrate Security Analysis into Source Management Process activity that requests more effort due to the intensive source code integration, however being more effective in terms of source code security risks identification.

The Verification phase (see Figure 4-7) comes to evaluate the source-code produced by the Implementation phase. This phase starts with the Identify, Implement and Perform Security Tests activity that is mandatory.

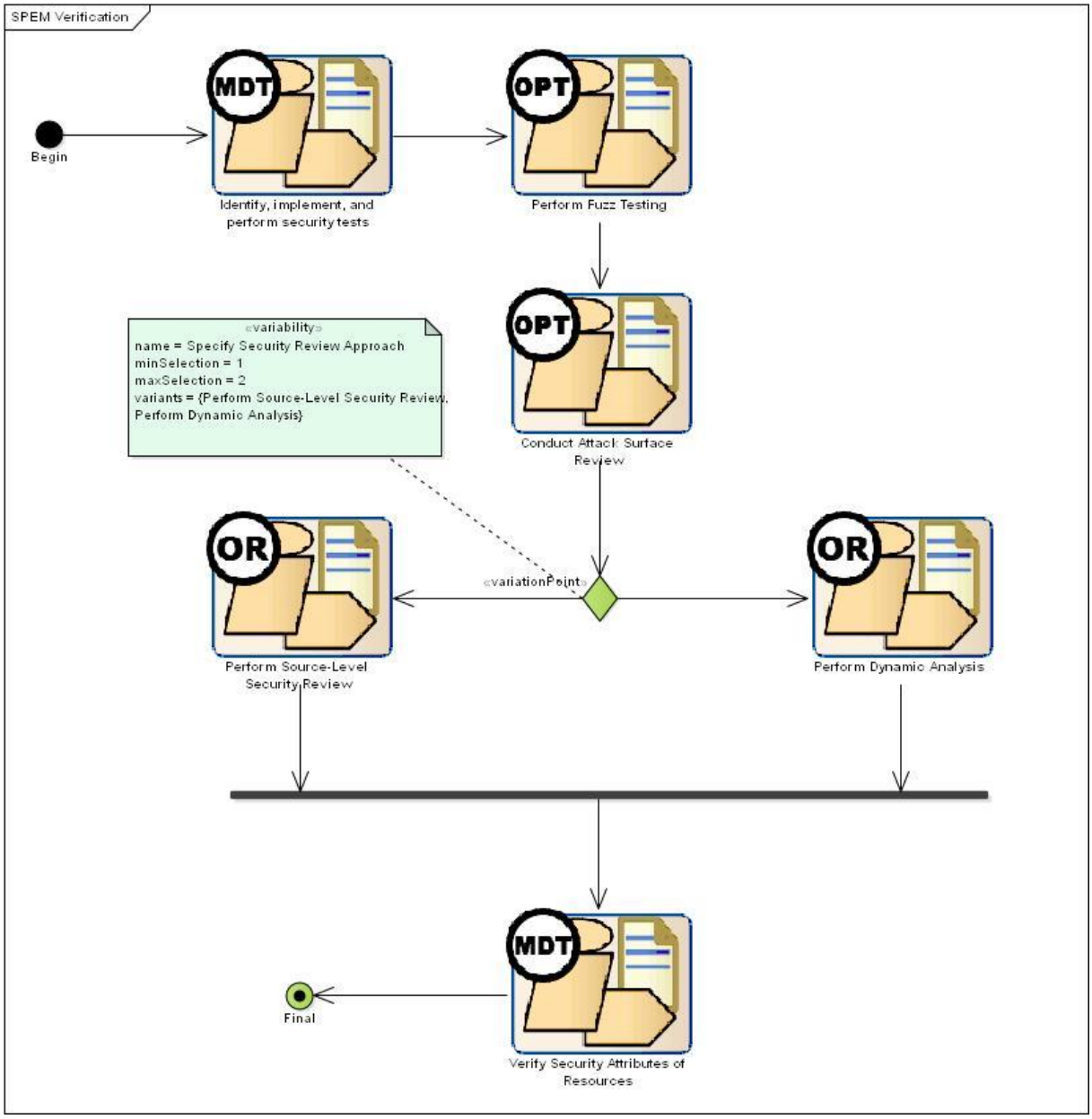

Figure 4-7. Verification phase overview (Author).

There is a variation point in the Verification phase. It is possible to execute the Perform Source-Level Security Review or the Perform Dynamic Analysis activities. As it is an “OR” condition, the project team can perform both or just one of them.

The last SPrL phase is the Release that contains the activities (see Figure 4-8) that support the final software release produced during the process development. This

phase assures that the solution security and provide the necessary guidance to fix the potential risks found during the entire process.

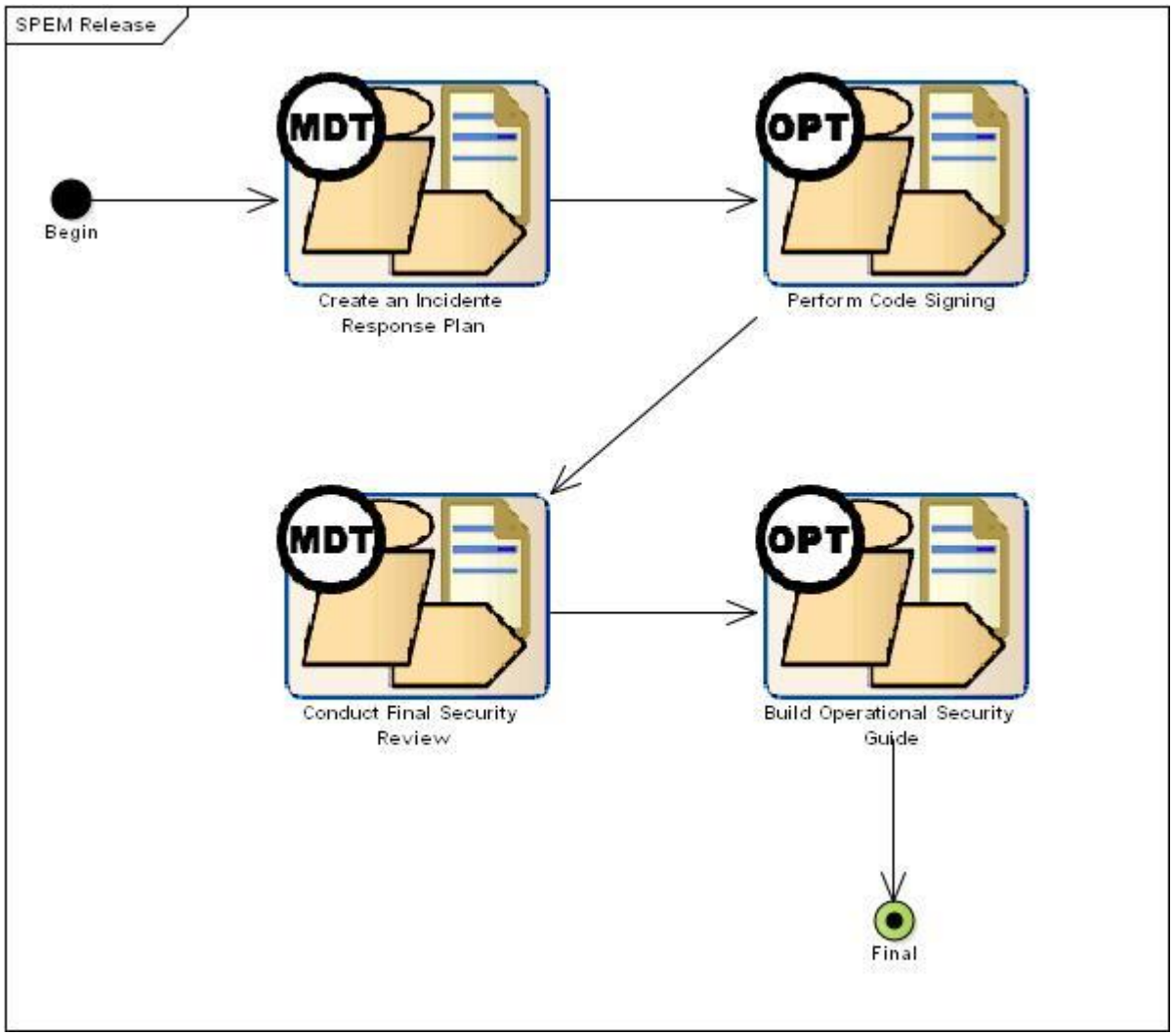

Figure 4-8. Release phase overview (Author).

This section presented the secure SPrL phases and their activities. The next section presents the evaluation of this proposal.

4.2.4. Evaluate the SPrL implementation

The implemented secure SPrL was evaluated in this first improvement cycle using as reference some candidate software development projects. The evaluation goal was to try to simulate the use of the SPrL instances in these projects. Due to the projects high complexity, it is necessary to observe their main characteristics (see Table 4-2).

Table 4.4. Selected projects to the research evaluation (Author).

Projects	Life-Cycle	Estimated Size (in hours)	Project Description
Project 01	Waterfall	1920	Mobile application to support external vendors to offer credit services. The project uses third party components and internal systems to exchange information from the customer database and the credit score system.
Project 02	RUP	2240	A web application to manage remote access hard tokens request and delivery.
Project 03	RUP	1792	Web application with mobile interface to monitor stock options variation across multiple markets.
Project 04	RUP	2752	Web application to monitor exchange operations in middle-east markets.
Project 05	Waterfall	3264	Web application to manage marketing campaigns that offer product and services to special customers in South America.

After exercising the SPrL instantiation, the research group identified several difficulties to employ the secure development process activities in different project life cycles. In this case, the Tailoring Guide document proposed in this first improvement cycle should be reviewed to provide the necessary guidance when working with several life cycles, at least the classic Waterfall and the Iterative and Incremental approaches that can be used as reference in the most part of the company, such as RUP.

In addition, the research group also concluded that it was necessary to employ the proposed SPrL in real projects and monitor their development to achieve a real understanding about the secure process limitations and potential issues. However, before starting the projects development, the project managers should have some assistance when planning the projects after tailoring the projects' processes.

4.3. Second Improvement Cycle

The second improvement cycle was performed using as input the issues reported at the end of the first improvement cycle.

4.3.1. Data Gathering, Analysis and Feedback

The research group focused on working in the previous improvement cycle findings that were translated in the cycle goals:

1. Improve the Tailoring Guide to support life-cycle tailoring assistance.

2. Export the SPrL instances to the MS Project tool to work on the projects planning.
3. Develop two real projects with the proposed SPrL.

These improvement cycles goals implementation was planned as described in the next section.

4.3.2. Action Planning

To improve the Tailoring Guide document to support the life-cycle adaption, it was necessary to consider some conceptual aspects, such as process's phases and activities frequency of use. The EPF tool has this information attribute defined for each activity. In this case, all documented SPrL activities were classified with the appropriate frequency of use values: Planned, Repeatable, Multiple Occurrences, Ongoing, Event-driven and Optional (see Figure 4-9). The frequency of use is an important information that can drive to process's iterations definition when working with no waterfall life-cycles.

Presentation Name	Planned	Repeatable	Multiple Occurrences	Ongoing	Event-Driven	Optional
SecureIterativeIncremental	☑	☐	☐	☐	☐	☐
Iteration 1	☑	☑	☐	☐	☐	☐
Training	☑	☐	☐	☐	☐	☐
Core Security Training	☑	☐	☐	☐	☐	☐
Institute Security Awareness Program	☐	☐	☐	☐	☐	☑
Management	☑	☐	☐	☐	☐	☐
Monitor Security Metrics	☐	☐	☐	☑	☐	☐

Figure 4-9. Setting tasks frequency in the EPF tool (Author).

To assist the project managers to plan their projects after defining the processes instances, a MS Project file can be exported from the EPF tool with the process iterations, phases and activities as presented in Figure 4-10.

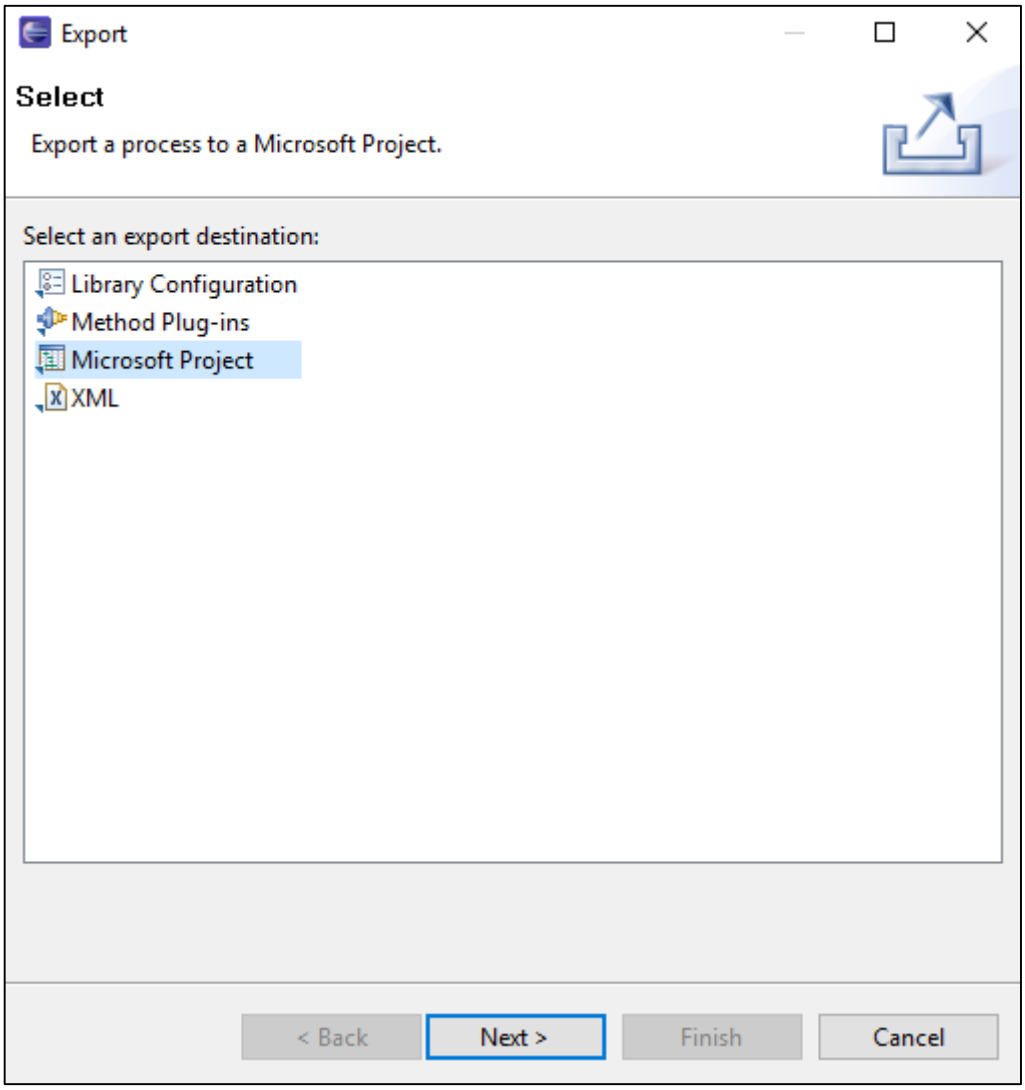

Figure 4-10. EPF tool support for MS Project format export (Author).

Using these mentioned EPF tool features, it was possible to proceed with the necessary SPrL review described in the next section.

4.3.3. Review the SPrL for secure development

The required actions for the second improvement cycle do not change the SPrL core structure in terms of activities flow and variation points definition. The goals are focused in the Tailoring Guide document adjustment that can be supported by the EPF tool which allows the specification of process iterations.

To simplify the iterations definition, the research groups created the structure presented in Figure 4-11, that was integrated to the secure SPrL specification.

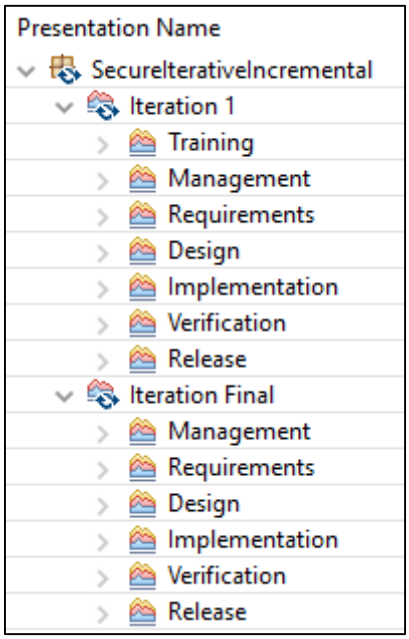

Figure 4-11. SPrL organization in iterations (Author).

Some phases or activities can be dismissed in some iterations, such as the Training phase that is performed just in the first process iteration. The tasks frequency information also supports the decision of dismiss some tasks when planning the process iterations. In this case, the project team can easily define a set of tasks for a specific iteration.

When working with waterfall life cycles, it is not necessary to consider process iterations as presented in Figure 4.12.

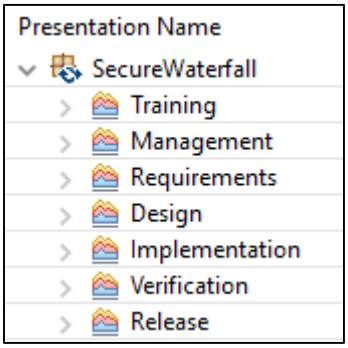

Figure 4-12. Waterfall life-cycle representation for SPrL (Author).

4.3.4. Evaluate the SPrL implementation

To complete the SPrL review, the research group employed the SPrL in two real projects. In this case, Project 01 (Waterfall) and Project 03 (Iterative-Incremental) that were previously described in Table 4.2. Their respective project managers exported

the SPrL work-breakdown tasks structure into the MS Project to manage and control the tasks and artifacts development.

The Project 01 was instantiated without the task Institute Security Awareness, once this task was already performed by the Security area 2 months before the project research. In this case, in the phase Training the only performed task was the Core Security Training that was offered to the Project 01 ‘s team members as a practical workshop, see Figure 4-13.

	Modo da	Nome da Tarefa	Duração	Início	Término
1		**Project 01 - Waterfall**	**106 dias**	**Seg 11/01/16**	**Sex 10/06/16**
2		**Training**	**5 dias**	**Seg 11/01/16**	**Sex 15/01/16**
3		Core Security Training	5 dias	Seg 11/01/16	Sex 15/01/16
4		**Management**	**101 dias**	**Seg 18/01/16**	**Sex 10/06/16**
5		Monitor Security Metrics	105 dias	Seg 18/01/16	Sex 10/06/16
6		**Requirements**	**28 dias**	**Seg 18/01/16**	**Sex 26/02/16**
7		Document Security Requirements	10 dias	Seg 18/01/16	Sex 29/01/16
8		Detail Misuse Cases	10 dias	Seg 01/02/16	Sex 12/02/16
9		Create Quality Gates	3 dias	Seg 15/02/16	Qua 17/02/16
10		Perform Risk Analysis	5 dias	Qui 18/02/16	Qua 24/02/16
11		Specify Operational Environment	2 dias	Qui 25/02/16	Sex 26/02/16

Figure 4-13. Project 01 planning with Training and Requirement phases (Author).

In terms of Requirements, the project team selected the option of performing the Detail Misuse Cases task. In the SPrL webpage, generated by the EPF Tool, it is possible to observe that after performing the Document Security Requirements task, the project team had 2 options of tasks to perform, as presented in Figure 4-14. These tasks specification are available in Appendices B.

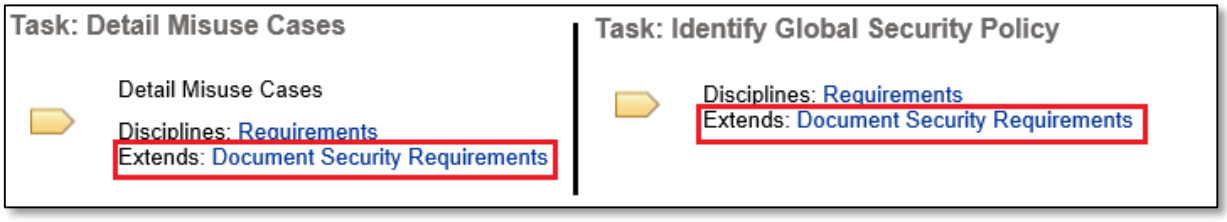

Figure 4-14. Variation point options in the Requirements phase (Author).

In the Project 01's Design phase, the project team had one variation point to define the design review approach. In this point, the project team option was performing the Establish Design Requirements task, as presented in Figure 4-15.

	Modo da	Nome da Tarefa	Duração	Início	Término
12		**Design**	**24 dias**	**Seg 29/02/16**	**Sex 01/04/16**
13		Research and Assess Security Posture of Technology Solutions	5 dias	Seg 29/02/16	Sex 04/03/16
14		Specify Database Security Configuration	3 dias	Seg 07/03/16	Qua 09/03/16
15		Establish Design Requirements	3 dias	Qui 10/03/16	Seg 14/03/16
16		Annotate Class Designs with Security Properties	5 dias	Ter 15/03/16	Seg 21/03/16
17		Apply Security Principles to Design	4 dias	Ter 22/03/16	Sex 25/03/16
18		Perform Risk Analysis	5 dias	Seg 28/03/16	Sex 01/04/16

Figure 4-15. Variation point option in the Design phase (Author).

The Project 01's Implementation phase, was conducted as presented in Figure 4-16. Where the variation point selected option was to Perform Static Analysis task, instead of using the Integrate Security Analysis into the Source Management Process task that would request more tools and specialized analysts to be successful implemented by the project team. The last variation point in this project planning was the Perform Static Analysis task instead of the Perform Dynamic Analysis. The option of not performing a dynamic analysis, was took by the project team due to the project potential risk analysis with low risk value.

	Modo da	Nome da Tarefa	Duração	Início	Término
19		**Implementation**	**19 dias**	**Seg 04/04/16**	**Sex 29/04/16**
20		Implement and Elaborate Resource Policies and Security Technologies	4 dias	Seg 04/04/16	Qui 07/04/16
21		Deprecate Unsafe Functions	1 dia	Sex 08/04/16	Sex 08/04/16
22		Implement Interface Contracts	5 dias	Seg 11/04/16	Sex 15/04/16
23		Perform Static Analysis	10 dias	Seg 18/04/16	Sex 29/04/16
24		**Verification**	**19 dias**	**Seg 02/05/16**	**Qui 26/05/16**
25		Identify, Implement and Perform Security Tests	10 dias	Seg 02/05/16	Sex 13/05/16
26		Perform Attack Surface Analysis	3 dias	Seg 16/05/16	Qua 18/05/16
27		Perform Static Analysis	2 dias	Qui 19/05/16	Sex 20/05/16
28		Verify Security Attributes of Resources	4 dias	Seg 23/05/16	Qui 26/05/16
29		**Release**	**11 dias**	**Sex 27/05/16**	**Sex 10/06/16**
30		Create an Incident Response Plan	5 dias	Sex 27/05/16	Qui 02/06/16
31		Perform Code Signing	2 dias	Sex 03/06/16	Seg 06/06/16
32		Conduct Final Security Review	1 dia	Ter 07/06/16	Ter 07/06/16

Figure 4-16. Project 01's Implementation and Evaluation phases planning (Author).

No relevant issues were reported by the project teams in terms of process adaption. However, the research group identified improvements opportunities in process maturity assessment that can be explored in further works. The full projects planning are available in Appendices C.

CHAPTER 5 - RESEARCH EVALUATION

This is not the end. It is not even the beginning of the end.
But it is, perhaps, the end of the beginning
- Winston Churchill

This chapter describes the research development and its results after using the action-research method.

5.1. SERVQUAL Questionnaire

After the second improvement cycle, the research groups applied a questionnaire based on the SERVQUAL approach to evaluate the proposed SPrL. Defined by Parasuraman in 1988, as an assessment method to evaluate services quality, the SERVQUAL approach uses a multi-item scale to evaluate factors that can impact the services and processes users' perspective.

Considering the SPrL's usability and utility characteristics, a set of 11 process quality factors were defined. For each factor, the users could inform their perceived level in a scale from 1 to 9. In addition, users could inform their minimum acceptable and maximum desirable levels of acceptance for each factor using the same scale. The goal is to analyze the variance among the perceived values and the expected minimum and maximum levels. The identified factors were:

- F01 - The SPrL purpose is clearly understood.
- F02 - The SPrL instantiation is an easy task.
- F03 - The SPrL can be easily extended (inclusion of new activities, roles, etc).
- F04 - The SPrL variation points are well documented.
- F05 - The SPrL variation points are relevant.
- F06 - The SPrL variation points are enough.
- F07 - The SPrL common activities are well documented.
- F08 - The SPrL common activities are relevant.
- F09 - The SPrL common activities are enough.

- F10 - The SPrL content is complete enough to support all secure development activities.
- F11 - The SPrL contributes to the secure software development.

In terms of the SPrL development support, two factors regarding the Action-Research and the EPF tool were also defined:

- F12 - The Action-Research method was a successful approach to the SPrL development.
- F13 - The EPF tool enabled the SPrL use, instead of the manual instantiation.

To complement the factors evaluation, two open questions were also elaborated to collect other users' perspectives.

- F14 - Do you recommend the use of the proposed SPrL?
- F15 - Do you have any suggestion to improve the proposed SPrL?

5.2. SERVQUAL Answers

The SERVQUAL questionnaire was answered by 15 of 18 SPrL users that had participated in the SPrL research and development. The answers were collected and organized in a single table, as presented in Table 5-1. Each factor is represented by its average value. See details in the Appendices A.

Table 5-1. Score average for each evaluation factor.

Levels	F01	F02	F03	F04	F05	F06	F07	F08	F09	F10	F11	F12	F13	Avg
Minimum acceptable	6,20	6,13	6,20	6,20	6,27	6,20	6,20	6,20	6,20	6,27	6,27	6,00	6,00	6,18
Maximum desirable	8,80	8,80	8,67	8,67	8,87	8,80	8,87	8,87	8,87	8,80	8,93	8,73	8,87	8,81
Perceived	8,13	8,07	7,80	6,73	8,47	8,00	7,80	8,47	8,07	8,47	8,53	7,53	7,60	7,97

As the perceived value average achieved was 7,97 in relation of the maximum desirable value of 8,81, the overall analysis indicated that the SPrL achieved 91% of the maximum desirable score for all evaluation factors.

In Figure 5-1, it is possible to observe the factors and identify the most concerning factors.

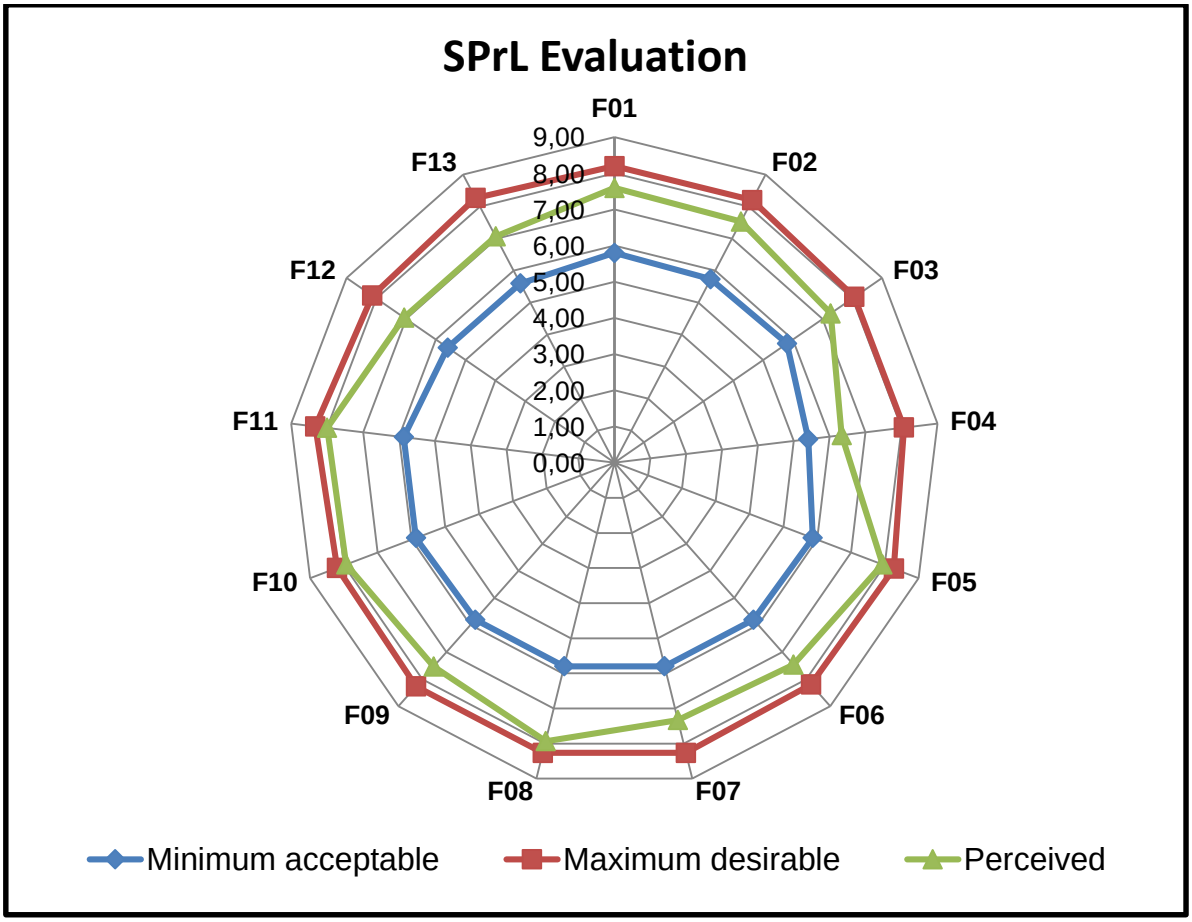

Figure 5-1. SPrL graphical overview (Author).

The factors F04, F12 and F13 had the lowest score values. The research group analyzed these results that were also commented in the open questions:

- F04: the SPrL variation points are not well documented, in fact the variation points details could be better specified, once they were described using just the SmartySPEM profile and some comments about them were presented in the Tailoring Guide document.
- F12: the Action-Research method was a concern for part of the users that were not used to work with this proposed approach. The process users were not familiar with the use of research methods in an organizational environment.
- F13: the EPF tool was not a consensus by the users. As the tool have not an extensive support by the Eclipse project supplier, some issues regarding the tool performance and bugs were reported by the users, such as problems when exporting to the MS Project application, generating the HTML process website, etc. As an alternative, other commercial SPEM based tools could be employed to support the SPrL use.

The factors that obtained the highest score levels, were the F05, F08, F10 and F11.

- F05: indicates that the SPrL variation points are relevant. Even, with concerns about their documentation, as previously described
- F08: the SPrL users appointed also how the SPrL common activities are relevant. In this case, the selection and merge of activities performed in the first improvement cycle was successful.
- F10: the SPrL content is complete enough to support all secure development activities. To the users, activities, roles, phases and artifacts are complete enough, supporting the SPrL structure and scope.
- F11: this factor high score indicated that the proposed SPrL is a real contribution to the secure software development.

About the open questions, the summary of answers was as following:

- F14: all process users recommended the use of the proposed SPrL.
- F15: in terms of suggestions to improve the SPrL, just 7 of 15 users provided some answer. There are 3 answers suggesting improvements in the variation points documentation. More 3 answer suggesting the use of other tools to support the SPrL instantiation, such as the IBM Rational Method Composer. And just 1 user provide some suggestion about the use of other research methods instead of the Action-Research approach.

The SERVQUAL questionnaire provided an evaluation based on the process user perception. These findings can contribute for further analysis and SPrL improvement cycles planning.

CHAPTER 6 - CONCLUSION

This is the end. My only friend, the end.
- The Doors

The research conclusion is described in this chapter, including the relevance of the study, research contribution, limitations e future works.

7.1. Relevance of the Study

The definition of a SPrL for secure development represents a contribution to the software development area, once there is a lack of similar studies that could support the tailoring of security engineering activities into the software development process.

The main secure development processes such as the CLASP, Microsoft SDL and Touchpoints provide method contents that enable the secure software development. However, these processes do not provide the necessary guidance to perform the tailoring of security aspects in different organizations contexts.

7.2. Research Contribution

The main contribution of this research project is the proposed SPrL that can be applied in different organizations and can be extended to enable new phases, activities, artifacts and roles when necessary.

The use of the Action-Research method facilitated the interaction among the research group, process users and organization senior management that supported the research development.

7.3. Research Limitations

The research was developed in a financial organization that provided a controlled environment and allowed the use of the SPrL in a set of real projects. However, it would be necessary to consider the use of the proposed SPrL in other environments and projects. In this case, more Action-Research improvement cycles could be performed by more specialized individuals.

The tools employed in this research were a concern to some process users. As the research did not have a budget to acquire licenses to use other tools, it was not possible to try alternative ones.

It was not possible to conduct an experiment in order to assess if the product that was the result of the SPrL application has more secure elements than if it was developed using the old process. This was due to real conditions in which the research was conducted.

7.4. Future Works

As future works, the SPrL could be applied in more projects in different organizations. In these new contexts, it would be possible to explore the variation points specified in this research and even improve them to offer more flexibility to the proposed SPrL for secure development.

To assure the process quality, it is possible to discuss alternatives to define a capability and maturity model specific for secure development processes. This opportunity could be explored in a further research.

A detailed experiment using two different teams with similar backgrounds and similar context conditions shall be conducted in order to assess the results of the application of the defined SPrL.

REFERENCES

(AGRAWAL; KHAN, 2009) AGRAWAL, A.; KHAN, R. A. **Measuring the vulnerability of an object-oriented design**. Network Security, v. 2009, n. 10, p. 13–17, October 2009.

(AJILA; KABA, 2008) AJILA, S. A.; KABA, A. B. **Evolution support mechanisms for software product line process**. Journal of Systems and Software, v. 81, n. 10, p. 1784–1801, January 2008.

(ALOTAIBI; LIU, 2014) ALOTAIBI, Y.; LIU, F. **A novel secure business process modeling approach and its impact on business performance**. Information Sciences, v. 277, p. 375–395, February 2014.

(ALEGRÍA; BASTARRICA, 2012) ALEGRÍA, J. A. H.; BASTARRICA, M. C. **Building software process lines with CASPER**. In: 2012 INTERNATIONAL CONFERENCE ON SOFTWARE AND SYSTEM PROCESS, ICSSP 2012. **Proceedings...** Zurich, Switzerland: IEEE, 2012, p. 170–179.

(ARMBRUST et al., 2009) ARMBRUST, O.; KATAHIRA, M.; MIYAMOTO, Y.; MUNCH, J.; NAKAO, H.; OCAMPO, A. **Scoping Software Process Lines**. Software Process Improvement and Practice, v. 14, p. 181–197, May 2009.

(BACA; CARLSSON, 2011) BACA, D.; CARLSSON, B. **Agile development with security engineering activities**. In: PROCEEDING OF THE 2ND WORKSHOP ON SOFTWARE ENGINEERING FOR SENSOR NETWORK APPLICATIONS - SESENA ’11 **Proceedings...** Honolulu, HI: ACM Press, 2011, p.149 –158.

(BACHMANN; CLEMENTS, 2005) BACHMANN, F.; CLEMENTS, P. C. **Variability in software product lines.** (Technical Report CMU/SEI-2005-TR012), September 2005, 46p. Available at: <http:// http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=7675>. Accessed on: 31-may-15.

(BARTSCH, 2011) BARTSCH, S. **Practitioners’ Perspectives on Security in Agile Development**. In: 2011 SIXTH INTERNATIONAL CONFERENCE ON AVAILABILITY, RELIABILITY AND SECURITY. **Proceedings…**Vienna, Austria: IEEE, 2011, p. 479–84.

(CASTRO; CRESPO; GARCÍA, 2013) CASTRO, S. J. B; CRESPO, R. G.; GARCÍA, V. H. M. **Patterns of Software Development Process**. International Journal of Interactive Multimedia and Artificial Intelligence, v. 1, n. 4, p. 33, 2011.

(CHATTERJEE; GUPTA; DE, 2013) CHATTERJEE, K.; GUPTA, D.; DE, A. **A framework for development of secure software.** CSI Transactions on ICT, v. 1, p. 143–157, June 2013.

(CHEN; BABAR, 2011) CHEN, L.; ALI BABAR, M. **A systematic review of evaluation of variability management approaches in software product lines**. Information and Software Technology, v. 53, n. 4, p. 344–362, 2011.

(CLEMENTS; NORTHROP, 2002) P. Clements, L. Northrop. **Software Product Lines: Practices and Patterns**. Reading, MA: Addison-Wesley, 2002.

(MCGREGOR, 2004) MCGREGOR, J. D. **Software Product Lines**. Journal of Object Technology, v. 3714, n. 3, p. 65–74, 2004.

(COUGHLAN; COGHLAN, 2002) COUGHLAN, P.; COGHLAN, D. **Action research for operations management.** International Journal of Operations & Production Management, v. 22, n. 2, p. 220 – 240, 2002.

(ELAHI; YU; ZANNONE, 2010) ELAHI, G.; YU, E.; ZANNONE, N. **A vulnerability-centric requirements engineering framework: Analyzing security attacks, countermeasures, and requirements based on vulnerabilities**. Requirements Engineering, v. 15, p. 41–62, 2010.

(EL-ATTAR, 2014) EL-ATTAR, M. **From misuse cases to mal-activity diagrams: Bridging the gap between functional security analysis and design**. Software and Systems Modeling, v. 13, p. 173–190, 2014.

(FABIAN et al,, 2010) FABIAN, B.; GÜRSES, S.; HEISEL, M.; SANTEN, T.; SCHMIDT, H. **A comparison of security requirements engineering methods**. Requirements Engineering, v. 15, p. 7–40, 2010.

(FUTCHER; SOLMS, 2008) FUTCHER, L.; SOLMS, R. **Guidelines for Secure Software Development**. In: CONFERENCE OF SOUTH AFRICAN INSTITUTE OF COMPUTER SCIENTISTS AND INFORMATION TECHNOLOGISTS - SAICSIT 2008, **Proceedings...** Wilderness, South Africa: 2008, p.56–65.

(HARRIS, 2008) HARRIS, S. ***All in One Cissp – Exam Guide Fourth Edition***. New York, NY: McGraw Hill, 2008. 748 p.

(HOUMB et al., 2010) HOUMB, S. H.; ISLAM, S.; KNAUSS, E.; JÜRJENS, J.; SCHNEIDER, K. **Eliciting security requirements and tracing them to design: An integration of Common Criteria, heuristics, and UMLsec**. Requirements Engineering, v. 15, p. 63–93, 2010.

(HURTADO, 2013) HURTADO, J. A.; BASTARRICA, M. C.; OCHOA, S. F.; SIMMONDS, J. **MDE software process lines in small companies**. Journal of Systems and Software, v. 86, n. 5, p. 1153–1171, 2013.

(ISO/IEC, 2004) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 13335-1:2004** Information technology - Security techniques – Management of Information and communications technologic security. ISO/IEC, 2004.

(ISO/IEC, 2005) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC FDIS 17799:2005** Information technology - Security techniques – Code of practice for information security management. ISO/IEC, 2005.

(ISO/IEC, 2008a) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 12207:2008** Standards Catalogue - Systems and software engineering -- Software life cycle processes. ISO/IEC, 2008. Available: <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43447>. Accessed: 15-Feb-15.

(ISO/IEC, 2008b) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 21827:2008** Standards Catalogue - Systems Security Engineering - Capability Maturity Model® (SSE-CMM®). ISO/IEC, 2008. Available: <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=44716>. Accessed: 15-Feb-15.

(ISO/IEC, 2009) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 15408:** Part 1 - Information technology - Security techniques - Evaluation criteria for IT security Information technology. ISO/IEC, 2009.

(ISO/IEC, 2011) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 27034:** Part 1 - Information technology - Security techniques – Application security – Overview and concepts. ISO/IEC, 2011. Available: <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=44378>. Accessed: 18-Feb-15.

(ISO/IEC, 2013) INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. **ISO/IEC 27001:** Information technology - Security techniques – Information Security Management System - Requirements. ISO/IEC, 2013. Available: <https://www.iso.org/obp/ui/#iso:std:iso-iec:27001:ed-2:v1:en>. Accessed: 18-Feb-15.

(KHAN; MUSTAFA, 2009) KHAN, R. A.; MUSTAFA, K. **From threat to security indexing: a causal chain**. Computer Fraud and Security, v. 2009, n. 5, p. 9–12, 2009.

(KITCHENHAM et al., 2009) KITCHENHAM, B.; BRERETON, O. P.; BUDGEN, D.; TURNER, M.; BAILEY, J.; LINKMAN, S. **Systematic literature reviews in software engineering - A systematic literature review**. Information and Software Technology, v. 51, n. 1, p. 7–15, 2009.

(KOTHARI, 2004) Kothari, C.R. **Research Methodology: Methods and Techniques**. Daryaganj, Delhi, IND: New Age International, 2004. ProQuest ebrary. Web. 5 March 2015.

(LORENZ et al., 2014) LORENZ, W. G.; BRASIL, M. B.; FONTOURA, L. M.; PEREIRA, G. V. **Activity-based Software Process Lines Tailoring**. International Journal of Software Engineering and Knowledge Engineering, v. 24, n. 9, p. 26, 2014.

(LORIN, 1985) LORIN, H. **Emerging security requirements**. Computer Communications, v. 8, n. 6, p. 293–298, 1985.

(MELLADO et al., 2010) MELLADO, D.; BLANCO, C.; SÁNCHEZ, L. E.; FERNÁNDEZ-MEDINA, E. **A systematic review of security requirements engineering**. Computer Standards & Interfaces, v. 32, p. 153-165, February 2010.

(MELLADO; MOURATIDIS; FERNANDEZ-MEDINA, 2014) MELLADO, D.; MOURATIDIS, H.; FERNÁNDEZ-MEDINA, E. **Secure Tropos framework for software product lines requirements engineering**. Computer Standards & Interfaces, v. 36, n. 4, p. 711–722, 2014.

(MELO, 2008) Melo, L. P. **Proposta de Metodologia de Gestão de Risco em Ambientes Corporativos na Área de TI**. Dissertação de Mestrado em Engenharia Elétrica. Universidade de Brasília, 2008.

(MOUGOUEI; SANI; ALMASI, 2013) MOUGOUEI, D.; FAZLIDA, N.; SANI, M.; ALMASI, M. M. **S-Scrum: A secure methodology for agile development of web services**. The World of Computer Science and Information Technology Journal (WSCIT), v. 3, n. 1, p. 15–19, 2013.

(NEVES et al., 2015) Neves, L.; Borba, P.; Alves, V.; Turnes, L.; Teixeira, L.; Sena, D.; Kulesza, U. **Safe Evolution Templates for Software Product Lines**. Journal of Systems and Software, April 2015.

(NIST, 2010) NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, U.S. DEPARTMENT OF COMMERCE. **Guide for Applying the Risk Management Framework to Federal Information Systems (NIST Special Publication 800-37)**. Gaithersburg: 2010.

(NIST, 2011) NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, U.S. DEPARTMENT OF COMMERCE. **Managing Information Security Risk – Organization, Mission, and Information System View (NIST Special Publication 800-39)**. Gaithersburg: 2011.

(NIST, 2012) NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, U.S. DEPARTMENT OF COMMERCE. **Guide for Conducting Risk Assessment (NIST Special Publication 800-30)**. Gaithersburg: 2012.

(OTHMANE et al., 2014) OTHMANE, L.; ANGIN, P.; WEFFERS, H.; BHARGAVA, B. **Extending the Agile Development Approach to Develop Acceptably Secure Software**. Journal of IEEE Transactions on Dependable and Secure Computing, v. XX, n. XX, p. 1–14, 2014.

(OWASP, 2014) OPEN WEB APPLICATION SECURITY PROJECT. OWASP: the free and open software security community. Available: <https://www.owasp.org>. Accessed: 24-Feb-15.

(PETERS; PEDRYCZ, 2001) J. Peters, W. Pedrycz. **Software Engineering: Theory and Practices**. São Paulo, SP: Ed. Campus, 2001, 29 p.

(POHL; METZGER, 2006) POHL, K.; METZGER, A. **Variability Management in Software Product Line Engineering**. In: PROCEEDING OF THE 28TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING ICSE 06. **Proceedings…** Shanghai, China: 2006, p. 1049.

(PONSARD et al., 2007) PONSARD, C.; MASSONET, P.; MOLDEREZ, J. F.; et al. **Early verification and validation of mission critical systems**. Formal Methods in System Design, v. 30, p. 233–247, December 2007.

(POTTER, 2009) POTTER, B. **Microsoft SDL Threat Modelling Tool**. Network Security, v. 2009, n. 1, p. 15–18, January 2009.

(POPP et al., 2003) POPP, G.; JURJENS, J.; WIMMEL, G.; BREU, R. **Security-critical system development with extended use cases**. In: TENTH ASIA-PACIFIC SOFTWARE ENGINEERING CONFERENCE, **Proceedings…** Chiang Mai, Thailand: IEEE, 2003, p. 1-10.

(REAVIS, 2013) Reavis Consulting Group. **The emergence of software security standards: ISO / IEC 27034-1 : 2011 and your organization**. May 2013. Available at <http://www.microsoft.com/global/eu/RenderingAssets/pdf/The%20emergence%20of%20software%20security%20standards.pdf >. Accessed on 31-05-2015.

ROUILLE, E.; COMBEMALE, B.; BARAIS, O.; TOUZET, D.; JEZEQUEL, J. M. **Improving reusability in software process lines**. In: PROCEEDINGS - 39TH EUROMICRO CONFERENCE SERIES ON SOFTWARE ENGINEERING AND ADVANCED APPLICATIONS, SEAA 2013. **Proceedings…** Santander, Spain: 2013, p. 90–93.

(SCHRAMM; DOHRMANN; KUHTMANN, 2015) Schramm J, Dohrmann P, Kuhrmann M. **Development of Flexible Software Process Lines with Variability Operations : A Longitudinal Case Study**. In: INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING - EASE ’15. **Proceedings…** Nanjing, China: 2015.

(SIPONEN; BASKERVILLE; KUIVALAINEN, 2005) SIPONEN, M.; BASKERVILLE, R.; KUIVALAINEN, T. **Integrating Security into Agile Development Methods**. In: PROCEEDINGS OF THE 38TH ANNUAL HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES. **Proceedings…** Honolulu, HI: IEEE, 2005, p. 1-4.

(SPEM, 2008) Object Management Group. Software and Systems Process Engineering Meta-model Specification, version 2.0. April, 2008. Available at: http://www.omg.org/spec/SPEM/2.0/PDF. Accessed on: 31-May-15.

(STEWART, 2009) STEWART, M. **Managing Information Risk: A Director's Guide**. Cambs, GBR: IT Governance, 2009. Accessed in February 22, 2015. ProQuest ebrary.

(STEWART et al., 2012) STEWART, J. M.; CHAPPLE, M.; GIBSON, D. **CISSP: Certified Information Systems Security Professional Study Guide (6th Edition)**.

Somerset, NJ, USA: John Wiley & Sons, 2012. ProQuest ebrary. Accessed: 23-Feb-15.

(TERNITE, 2009) TERNITÉ, T. **Process lines : a product line approach designed for process model development**. In: EUROMICRO CONFERENCE ON SOFTWARE ENGINEERING AND ADVANCED APPLICATIONS PROCESS. **Proceedings…** Patras, Greece: 2009, p. 173–180.

(UZUNOV; FERNANDEZ; FALKNER, 2012) UZUNOV, A. V.; FERNANDEZ, E. B.; FALKNER, K. **Engineering Security into Distributed Systems: A Survey of Methodologies**. Journal of Universal Computer Science, v. 18, n. 20, p. 2920–3006, December 2012.

(WIN et al., 2009) WIN, B. D.; SCANDARIATO, R.; BUYENS, K.; GRÉGOIRE, J.; JOOSEN, W. **On the secure software development process: CLASP, SDL and Touchpoints compared**. Information and Software Technology, v. 51, p. 1152–1171, February 2009.

Appendices A – SERVQUAL Questionnaire

Factors		Levels	Scale									Average
			1	2	3	4	5	6	7	8	9	
SPrL Usability												
F01	The SPrL purpose is clearly understood.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	0	3	12	**8,80**
		Perceived	0	0	0	0	0	0	2	9	4	**8,13**
F02	The SPrL instantiation is an easy task.	Minimum acceptable	0	0	0	0	2	9	4	0	0	**6,13**
		Maximum desirable	0	0	0	0	0	0	0	3	12	**8,80**
		Perceived	0	0	0	0	0	0	4	6	5	**8,07**
F03	The SPrL can be easily extended (inclusion of new activities, roles, etc).	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	1	3	11	**8,67**
		Perceived	0	0	0	0	0	1	6	3	5	**7,80**
SPrL Utility												
F04	The SPrL variation points are well documented.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	2	1	12	**8,67**
		Perceived	0	0	0	0	0	3	6	4	1	**6,73**
F05	The SPrL variation points are relevant.	Minimum acceptable	0	0	0	0	2	7	6	0	0	**6,27**
		Maximum desirable	0	0	0	0	0	0	0	2	13	**8,87**
		Perceived	0	0	0	0	0	0	0	8	7	**8,47**
F06	The SPrL variation points are enough.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	1	1	13	**8,80**
		Perceived	0	0	0	0	0	0	3	9	3	**8,00**
F07	The SPrL common activities are well documented.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	0	2	13	**8,87**
		Perceived	0	0	0	0	0	0	6	6	3	**7,80**
F08	The SPrL common activities are relevant.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	0	2	13	**8,87**
		Perceived	0	0	0	0	0	0	1	6	8	**8,47**
F09	The SPrL common activities are enough.	Minimum acceptable	0	0	0	0	2	8	5	0	0	**6,20**
		Maximum desirable	0	0	0	0	0	0	0	2	13	**8,87**
		Perceived	0	0	0	0	0	0	5	4	6	**8,07**
F10		Minimum acceptable	0	0	0	0	2	7	6	0	0	**6,27**

	The SPrL content is complete enough to support all secure development activities.	Maximum desirable	0	0	0	0	0	0	1	1	13	**8,80**
		Perceived	0	0	0	0	0	0	1	6	8	**8,47**
F11	The SPrL contributes to the secure software development.	Minimum acceptable	0	0	0	0	2	7	6	0	0	**6,27**
		Maximum desirable	0	0	0	0	0	0	0	1	14	**8,93**
		Perceived	0	0	0	0	0	0	0	7	8	**8,53333**
Action-Research Development												
F12	The Action-Research method was a successful approach to the SPrL development.	Minimum acceptable	0	0	0	0	3	9	3	0	0	**6,00**
		Maximum desirable	0	0	0	0	0	0	0	4	11	**8,73**
		Perceived	0	0	0	0	0	1	7	5	2	**7,53**
EPF Tool												
F13	The EPF tool enabled the SPrL use, instead of the manual instatiation.	Minimum acceptable	0	0	0	0	3	9	3	0	0	**6,00**
		Maximum desirable	0	0	0	0	0	0	0	2	13	**8,87**
		Perceived	0	0	0	0	0	0	8	5	2	**7,60**

Appendices B – SPrL Tasks Specification

Appendices C – Projects Planning