Formalization of Secure Service Oriented Product Line

Ines Achour, Lamia Labed and Henda Ben Ghezala
Department of Computer Science, RIADI-GDL, ENSI, Manouba, Tunisia

Keywords: Secure Service Oriented Product Line, Software Security, Secure Domain Engineering, Secure Application Engineering, Map.

Abstract: In this work, we focus on the SOPL approach (Service Oriented Product Line) which can be used in various domains where SOA based applications are needed such as e/m government, e-business, e-learning and so on. This approach is a combination of Service-Oriented Architecture (SOA) and Software Product Line (SPL). Ensure secure services are vital in order to establish trust between users and service providers. In this context, we aim to propose guidelines for using Secure SOPL which process leads to produce secure service-oriented applications. In fact, with the diversity of the means that allow us to perform security activities, the use of Secure SOPL is difficult especially for developers whose lack experience in the security software, SPL and SOA fields which are the basis the Secure SOPL. Thus, we choose the Map formalism which is a decision-oriented model to formalize the two phases of our Secure SOPL.

1 INTRODUCTION

Nowadays, companies rely on distributed architectures and more specifically on the architecture based services namely SOA (Service Oriented Architecture). This type of architecture is a solution to problems of integration and interoperability within services. In addition, companies hope satisfying customers' needs such as reducing cost, effort and time to market of software products. These advantages are offered by SOPL approach (Service Oriented Product Line) which is a combination of the key concepts of SOA and those of SPL (Software Product Line). SOPL enhance the systematic reuse of services but doesn't offer solutions to security problems especially as we live in a connected and in an opened world. In (Achour et al., 2015) we have integrated security activities throughout SOPL development cycle. The underlying idea is that the improvement of software products must go through improving their development process (Finkelstein et al., 1994). This leads us to propose Secure SOPL: a development process that integrates security in the SOPL process and that aims to produce secured software products based on SOA. However, the variety of means (methods, tools, etc.) that allow to perform security activities (that we have introduced on Secure SOPL) and the immaturity of SOPL approach make it difficult to implement Secure SOPL. Also, the security activities appear as just listing the existing techniques and tools. All these reasons lead us to thought on formalizing the Secure SOPL based on a decision-oriented approach. Our proposal must help developers, specifically if they are non experimented in security and/or SOA and/or or PLE which are in the core of secure SPL. The help consists in the choice of the activity to proceed and in the choice of the mean to perform this activity. The Map (Rolland et al., 1999) process meta-model is a decision-oriented process and it seems to best fit our needs. In fact, the Map formalism permits to represent the different steps and activities of the Secure SOPL as intentions and the means (guidelines) to use to perform those as strategies. The Map aims at flexibility: it provides a view that does not impose activities to take up, but enhances what can be done (next) and how it can be done.

In this paper, we present in section 2 an overview of the Secure SOPL process. In section 3, we present the Map formalism and the Map relating to the realization of the two phases of Secure SOPL. Section 4 is reserved to give an overview of related works. Section 5 presents an illustrative example. Finally, section 6 summarizes our proposal and outlines our future work.

Achour I., Labed L. and Ben Ghezala H..
Formalization of Secure Service Oriented Product Line.
DOI: 10.5220/0005559100930100
In *Proceedings of the 10th International Conference on Software Engineering and Applications* (ICSOFT-EA-2015), pages 93-100
ISBN: 978-989-758-114-4

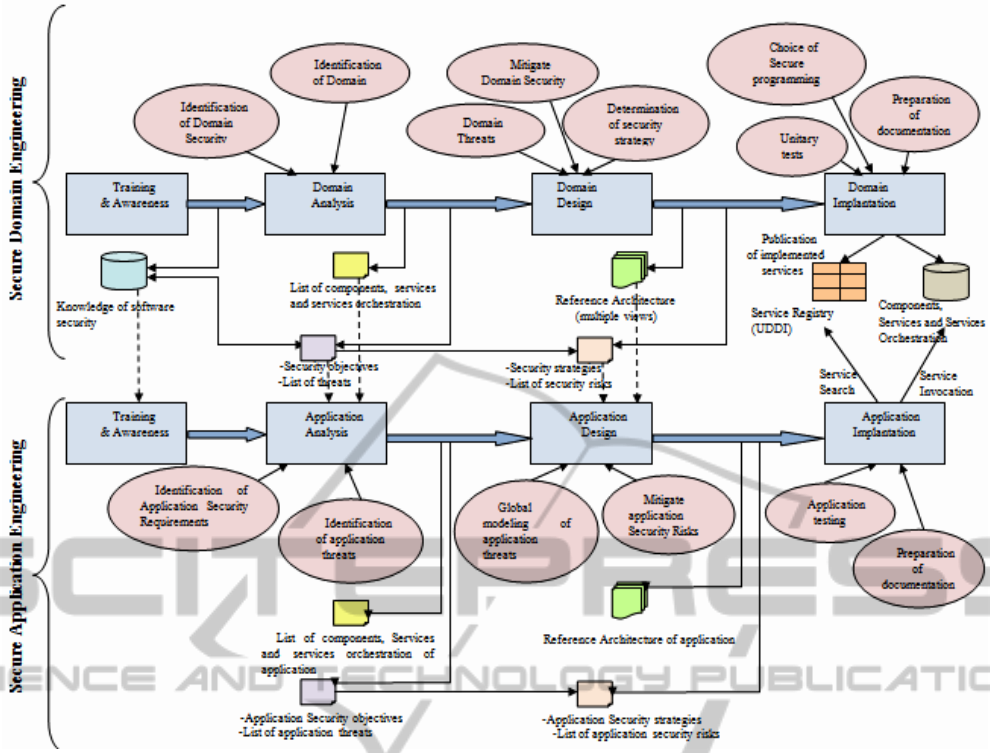

Figure 1: Map for Secure SOPL.

2 SECURE SOPL PROCESS

Service Oriented Product Line (SOPL) was introduced at the 11th edition of the International Conference SPLC (the 'Software Product Line Conference') in 2007. The SOPL approach is a combination of Software Product Line engineering and Service Oriented Architecture approach providing thus solutions to many common software problems as reuse and interoperability issues (Medeiros et al., 2009). It allows developing applications oriented SOA as Software Product Lines (SPL).

In order to add a security dimension to the SOPL approach, we have injected in (Achour et al., 2015) security activities on the SOPL process life cycle, mainly domain engineering (development for reuse) and application (development with reuse) phases (see Fig. 1), and proposed Secure SOPL. The Secure SOPL process begins with a domain engineering phase (Achour et al., 2015). This phase begins with a training and awareness step on which developers community for reuse are trained in security engineering basics. This will increase their awareness about the importance of the domain problems on which they operate (De Win et al., 2009), (Lipner, 2004), (McGraw et al., 2004), (Owasp Corporation et al., 2005). The developers use the feature model and the business process model as inputs (Berger et al., 2008), in order to produce during the domain analysis a list of components, services and composite services candidates for reuse. Also, to reinforce security, they identify the domain security requirements by several means, we quote: Abuse Case Diagram (McDermott et al., 1999) SQUARE method (Mead et al., 2005), RMF (Risk Management Framework) (McGraw et al., 2004) or standards such as the Common Criteria (Common Criteria - Part 1, 2,3, 2009).

During this step, developers mitigate security risks which consist on the synthesis and prioritization of risks (developers can use RMF (McGraw et al., 2004), STRIDE (Microsoft Corporation, 2015) or DREAD (Meier et al., 2003), etc) and the definition of the risk mitigation strategy (for example they can remove a feature, solve the problem, etc. (McGraw et al., 2004)).

We conclude this phase with the implementation of different components, services and services orchestration. The developers must respect a set of

practices (Howard, 2008) to improve the security code such as: choosing a programming language that offers more security (such J2EE (Krakowiak et al., 2006)), using security standards for service (for example using XML Encryption to provide confidentiality (Toms, 2009)), using code analysis tools (Security Innovation Corporation, 2004), etc. Also they must prepare documentation to the development community with reuse and they perform unitary tests for such components and services (we can use White-hat or Black-hat approaches (De Win et al., 2009)).

The application engineering phase (Achour et al., 2015) starts with training and awareness step on which developers community by reuse are trained in security engineering basics. Next, developers begin the application analysis step where they select components, services and services orchestration from the list identified in the domain engineering. The selection responds to the functionalities required by the developed application. For security needs, developers must select common security requirements and also common threats from the knowledge gained on the domain analysis and they must identify specific security requirements and specific threats which are not common for the applications of our line service and characterize the studied application. This step can be conducted by using the same means as those led on the domain analysis.

Then, the configuration and the specialization of selected components, services and composite services are performed in order to propose a specific architecture of the developed application. The threat modeling in the secure domain engineering phase is conducted for each component, services and services orchestration separately but here they are all integrated to form a specific application. So, the developers must conduct a global threat modeling approach. This can be done by STRIDE method (Microsoft Corporation, 2015).

Product construction concludes the application implantation step.

3 FORMALIZATION OF OUR SECURE SOPL

In this section, we start with the presentation of the Map formalism and we present the different formalization of Secure SOPL, Secure Domain Engineering and Secure Application Engineering according to this formalism.

3.1 Overview of Map Formalism

The Map (Rolland et al., 1999) process meta-model is classified in the category of processes of our interest. This is a directed graph in which the nodes are intentions and the arcs are strategies for ways to carry out the intentions.

Each Map has two special intentions, Start and Stop to start and stop respectively the process. Also, the Map is based on the section concept. A section is a triplet < Ii, Ij, Sij> which is a way to accomplish the intention targeted Ij from the source intention Ii using the strategy Sij. The Map contains a finite number of paths each requires a different way to achieve a final goal. The progress in the Map is provided by different types of directives, it offers:

- Directive of Intention Realization (DIR): It can provide operational means to meet the target for the section. It can be of three types: simple, tactical (plan or choice) or strategic.
- Directive of Strategy Selection (DSS): It determines the strategies that connect two intentions and helps to choose the most appropriate one depending on the situation.
- Directive of Intention Selection (DIS): It determines the intentions that follow a given one and helps to select one of them.

The two last ones are always tactic.

3.2 Formalization of Secure SOPL

For synthesizing the most important and essential security activities for secure development process, we based our study on (De Win et al., 2009), (McGraw et al., 2004), (Essafi et al., 2014), (Owasp Corporation et al., 2005), and (Lipner, 2004), we summarize in table 1 the principle activities and techniques ensuring security on the different software development process phases.

Based on this table we tried to formalize the Secure SOPL. In fact, Fig. 2 illustrates Secure SOPL modelized with the Map formalism. In fact, we progress to the “Develop Secure line service” intention based on Secure Domain Engineering which represents the first phase. Also, we progress to the “Develop Secure application based SOA” intention based on Secure Application Engineering which is the second phase. These phases are represented by two DIR for fulfilling intentions cited above.

Table 1: Activities and techniques ensuring security through the different phases of the software development process.

Phases of process development		Activities ensuring security	techniques
Training and awareness		-Plan regular courses which cover the latest security software issues -Promote sharing and communication artifacts -Assign a security adviser to the project	
Requirement engineering and Analysis		Identification of security requirements	-Abuse Case Diagram (McDermott et al., 1999) -SQUARE methods (Mead et al., 2005) -RMF (Risk Management Framework (McGraw et al., 2004)) -Standards (such as Common Criteria (Common Criteria - Part 1, 2,3, 2009)), etc.
		Identification of threats	-Bases of vulnerabilities (OSF, 2015) -STRIDE method (Microsoft Corporation, 2015), etc.
Design		Threats modeling	-Attack Tree (Essafi et al., 2014) -Threat Tree (Essafi et al., 2014) -Standards (such as Common Criteria (Common Criteria - Part 1, 2,3, 2009)) , etc.
		Mitigate Security Risks	-RMF (McGraw et al., 2004) -STRIDE method (Microsoft Corporation, 2015) -DREAD method (Meier et al., 2003), etc.
Implantation	Implementation	Secure programming language	-J2EE (Krakowiak et al., 2006) -using XML Encryption to provide confidentiality (Toms, 2009) -using code analysis tools (Security Innovation Corporation, 2004) , etc.
		Preparation of documentation	
	Testing	Unitary tests	-White-hat or -Black-hat approaches (De Win et al., 2009) , etc.

3.3 Formalization of Secure Domain Engineering

In this section we model the DIR that allows the development of a secure line. In fact, we represent on Fig. 3 all intentions that represent required steps and strategies to perform the first phase of our proposal.

We can view the intentions corresponding to the first phase of the classic SOPL: to form on security issues (relative to training and awareness), analyze domain, design domain and implement domain components and services (relative to domain implantation) and intentions corresponding to activities added to prevent security problems: identify domain security requirements, identify domain security threats, mitigate domain security risks and test developed components and services.

The DIR defines the way an intention could be released and could be simple, strategic or tactical. For example, “Using Feature Model” strategy, in <To form on security issue, Analyze domain, Using Feature Model > section, could be refined into: (a) “Using Feature Model”: Thus a <To form on security issue, Analyze domain, Using Feature Model > section which not requires the integration of non-functional attributes, and (b) “Using Extended Feature Model”: Thus a <To form on security issue, Analyze domain, Using Extended Feature Model > section which requires the integration of non-functional attributes.

The DIR (“Using Extended Feature Model”) mentioned above could be also refined. We notice that the construction of the extended feature model must integrate the non-functional attributes (Benavides, 2005) and the analysis of such models can be performed by Extend Flame tool (Achour et al., 2014).

3.4 Formalization of Secure Application Engineering

We refine on Fig. 4 the DIR that allows the development of a secure application based on SOA, which corresponds to the second phase of our Secure SOPL. We can view the intentions corresponding to the second phase of classic SOPL: to form on security issues (relative to training and awareness), analyze application, design application and implement secure application and intentions corresponding to activities added to prevent security problems: identify application security requirements, identify application security threats, mitigate application security risks and test application.

4 RELATED WORKS

Extensive work has been carried out on software security during the last few years, and there are several works that deal with security at the early stages of the development lifecycle, the same as Secure SOPL. We summarize some proposals particularly close in topic to ours and we explain their relation to our Secure SOPL.

SREP (Mellado et al., 2007) (Security Requirements Engineering Process) describes how to integrate security requirements into the software engineering process in a systematic and intuitive way.

In order to achieve this goal, the approach is based on the integration of the Common Criteria (CC) (ISO/IEC 15408) into the software lifecycle model.

However, SREP is only focused on the activities directly concerning security requirements elicitation and specification, while our proposal deals with all the lifecycle. Also our approach concerns the SOPL approach but SREP is applied to classical software development lifecycle.

SREPLine (Mellado et al., 2008), Security Requirements Engineering Process for software Product Lines (SREPPLine), which is a standard-based process that describes how to integrate security requirements into the software engineering process in a systematic and intuitive way, as well as

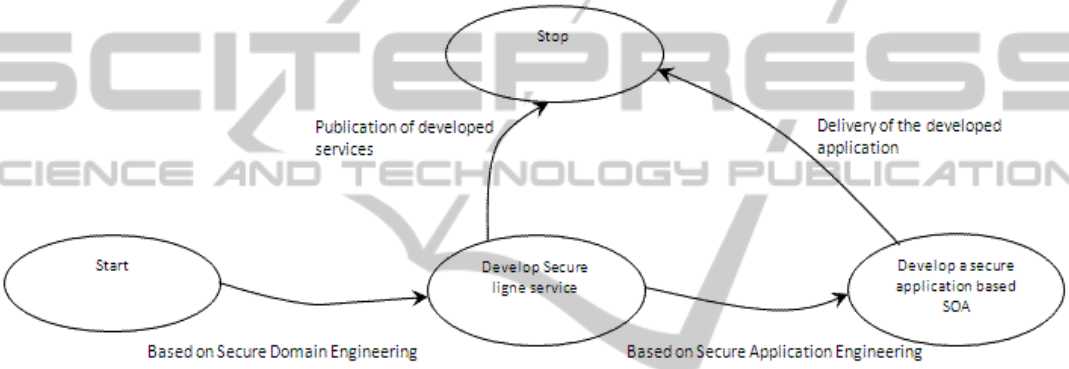

Figure 2 : Map for Secure SOPL.

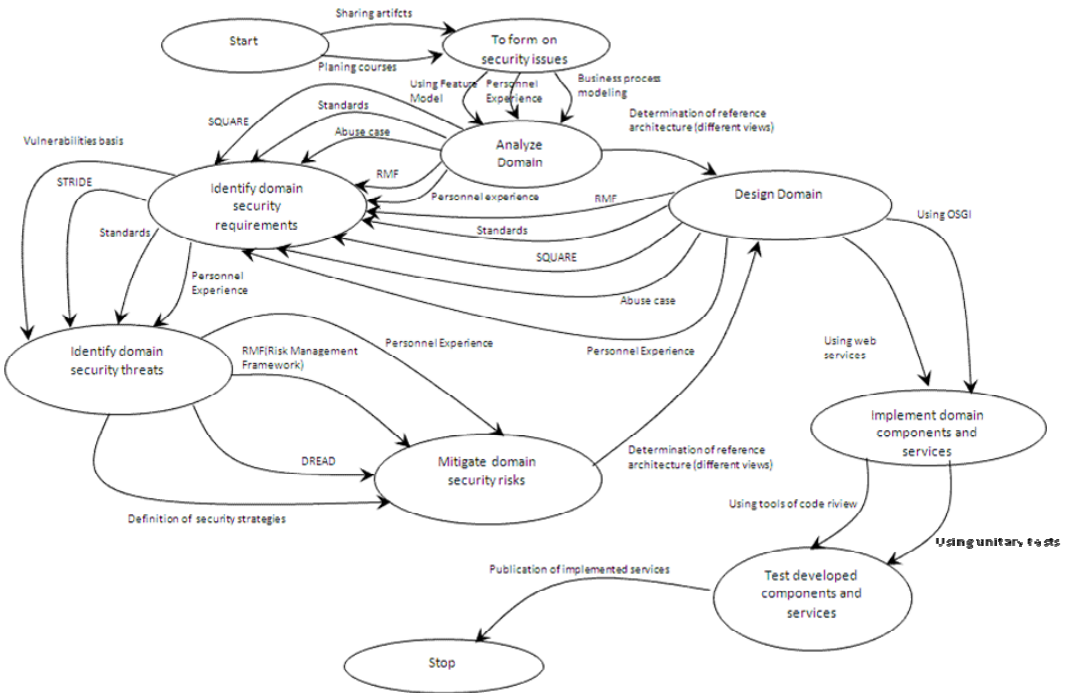

Figure 3 : Map for Secure Domain Engineering.

a simple integration with the rest of requirements and the different phases/processes of the SPL development lifecycle. Additionally, this process will facilitate the fulfillment of the IEEE 830:1998 standard. However, SREPLine is only focused on security requirements and it is applied to SPL, while our proposal deals with the two phases of SOPL approach.

S2D-ProM (Essafi et al., 2007), (Essafi et al., 2014) (Secure Software Development Process Model). This approach is a strategy oriented process model, with respect to the Map formalism, that allows to provide two level guidance: (a) a strategic guidance helping the developer to choose one among of existing techniques, methods, standards and best practices which are useful for producing secure software and (b) a tactical guidance on how to achieve his selection. However, S2D-ProM is applied to classical software development lifecycle, while Secure SOPL deals with SOPL approach.

5 ILLUSTRATIVE EXAMPLE

In order to show the feasibility of our Secure SOPL, we choose to study a range of governmental services offered by the Tunisian Ministry of the interior and local development (PRF Tunisian National Project, "Projet de Recherche Fédéré", where we are one of the parrtners) as the demand of National Identity Card (CIN), Passport and Bulletin n°3 (which is an official paper to mention if the person has corruption or not). We proceed by instantiating the different Maps. Due to the space limitations, we are going to present only some steps of the instantiation of the Secure Domain Engineering Map presented by Fig. 3. In order to analyze our domain which is an intention of the cited Map (see Fig. 3), after the study of the business requirements of our service line, we modeled the feature model of the online administration.

This model is based on a hierarchy of composition of characteristics (functional, non functional or parameters) where some branches are mandatory, some are optional, and others are mutually exclusive (Kang et al., 1990). It can show us commonality and variability of services. To achieve the intention “Identify domain security requirements", we choose to model our security requirements with abuse case diagram.

To accomplish “Identify domain security threats” intention, we use the OSVDB (Open Source

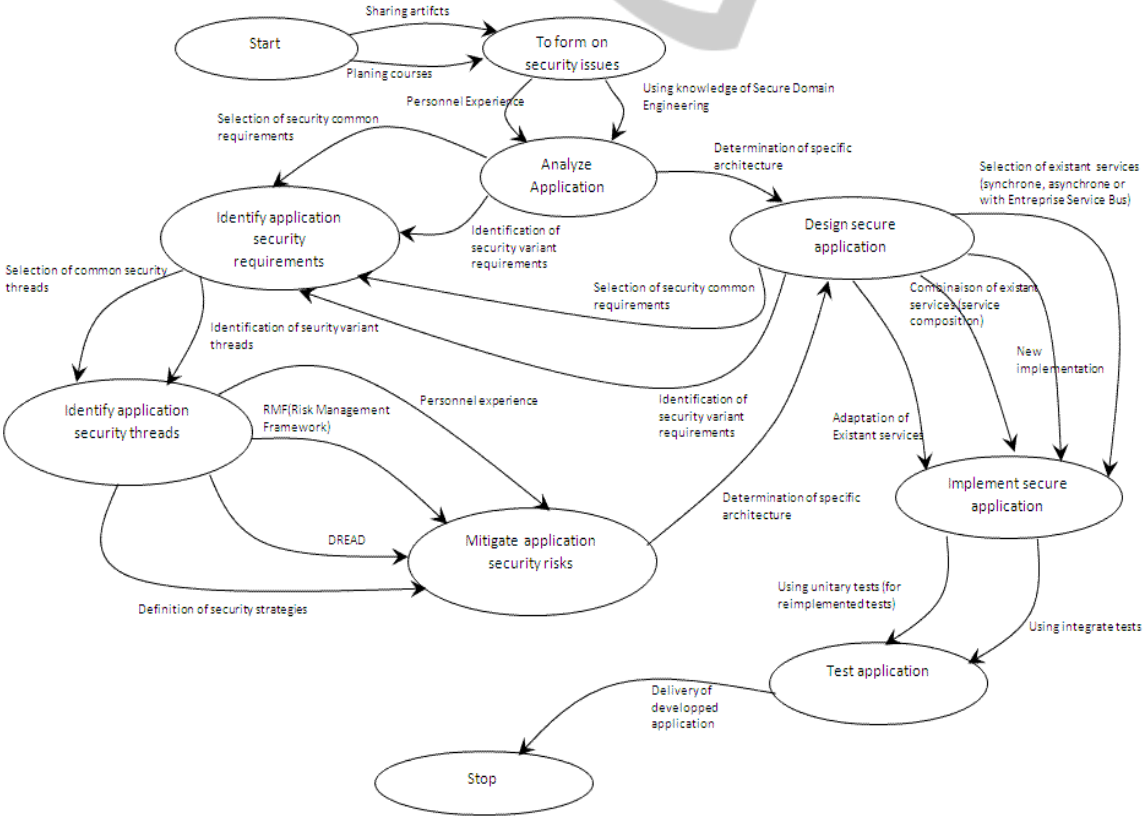

Figure 4: MAP for Secure Application Engineering.

Vulnerability Database) (OSF, 2015) to determine threats related to security requirements mentioned on our abuse case diagram.

To carry out the intention "Mitigate Domain Security Risks" we choose DREAD method (Meier et al., 2003) to evaluate risks.

Finally, to perform "Design domain" intention, we modeled our reference architecture.

6 CONCLUSIONS

The principle aim of this work is to propose guidelines for developers using Secure SOPL (Achour et al., 2015). We are based on Map formalism which permits to orient the developer for using security methods, concepts, standards and frameworks (such as RMF, STRIDE and Common Criteria) all well suited for given situations and contexts. This work aims to ensure the development of a product (based on SOA) by taking advantages of three concepts contributions: a large-scale reuse system (product line engineering), service-oriented architecture and software security. We presented an illustrative example related to a range of governmental services offered by the Tunisian Ministry of the interior and local development to show the feasibility of our proposal. Our perspectives are first to provide a tool which supports Secure SOPL. Second, we would like to validate the proposed approach in different contexts such as e-commerce, e-learning, etc.

REFERENCES

Achour, I., Labed, L., Ben Ghezala, H., 2014. Toward an Extended Tool for Analysis of Extended Feature Models, *In: the International Symposium on Network, Computer and Communications ISNCC'14*. Hammamet, Tunisia.

Achour, I., Labed, L., Ben Ghezala, H., 2015. Proposition of Secure Service Oriented Product Line, *In: the 6th International Conference on Information Systems and Economic IntelligenceSIIE'15*. Hammamet, Tunisia.

Benavides, D., Trinidad, P., Ruiz-cortés, A., 2005. Automated Reasoning on Feature Models. LNCS, Advanced Information Systems Engineering. *In: 17th International Conference, CAISE*.

Berger, T., Gunther, S., 2008. Service-Oriented Product Lines: Towards a Development Process and Feature Management Model for Web Services, *In: 12th International Software Product Line Conference (SPLC 2008)*, Limerick, Ireland.

Common Criteria for Information Technology Security Evaluation Norm ISO 15408 – Part 1: Introduction and general model – version 3.1 2009.

Common Criteria for Information Technology Security Evaluation Norm ISO 15408 – Part 2: Security functional requirements– version 3.1, 2009.

Common Criteria for Information Technology Security Evaluation Norm ISO 15408 – Part 3: Security assurance requirements– version 3.1, 2009.

De Win, B., Scandariato, R., Buyens, K., Grégoire, J., Joosen, W., 2009. On the secure software development process: CLASP, SDL and Touchpoints compared. *Information and Software Technology*, Vol. 51, No. 7, pp. 1152-1171.

Essafi, M., 2014. *Approche multi-démarches avec guidage flexible pour le développement de logiciels sécurisés,* Thesis, Manouba University.

Essafi, M., Labed L., Ben Ghezala, H., 2007. S2D-ProM: A Strategy Oriented Process Model for Secure Software Development, *In : the second International Conference on Software Engineering Advances (ICSEA 2007)*, Cap Esterel, French Riviera, France.

Finkelstein, A., Kramer, J., Nuseibeh, B., 1994. *Software Process Modelling and Technology*, Advanced Software Development Series, Research Studies Press/John Wiley &Sons..

Howard, M., 2008. *Microsoft Corporation: Fundamental practices for secure software development*, Stacy Simpson, SAFECode.

Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, S., 1990. *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Technical report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, Pittsburgh.

Krakowiak, S., Coupaye, T., Quema, V., Seinturier, L., Stefani, J., 2007. *Intergiciel et Construction d'Applications Réparties*.

Lipner, S., 2004. The Trustworthy Computing Security Development Lifecycle, Computer Security Applications Conference, 20th Annual Publication, ISSN: 1063-9527, ISBN: 0-7695-2252-1, pages 2-13.

McDermott, J., Fox, C., 1999. Using Abuse Case Models for Security Requirements Analysis, *In: 15th Annual Computer Security Applications Conference*, Phoenix, Arizona.

McGraw, G., 2004. Software Security: Building Security, *In. IEEE Computer Society, IEEE Security and Privacy*.

Mead, N. R., Hough, E. D., Stehney, T. R., 2005. *Security Quality Requirements Engineering (SQUARE) Methodology*, Technical report CMU/SEI-2005-TR-009, Carnegie Mellon University.

Medeiros, F., Romero, S., Santana, E., 2009. Towards an Approach for Service-Oriented Product Line Architectures. *In: 13th International Software Product Line Conference* (SPLC 2009), San Fransisco, CA, USA.

Meier, J. D., Mackman, A., Vasireddy, S., Dunner, M., Escamilla, R., Murukan, A., 2003. *Improving Web Application Security: Threats and Countermeasures*. Microsoft Corporation.

Mellado, D., Fernández-Medina, E., Piattini, M. , 2007. A common criteria based security requirements engineering process for the development of secure information systems, *Computer Standards and Interfaces* Volume 29 (2), pp 244–253.

Mellado, D., Fernández-Medina, E., Piattini, M., 2008. Towards security requirements management for software product lines: A security domain requirements engineering process, *Computer Standards & Interfaces* Volume 30, Issue 6, pp 361–371,.

Microsoft Corporation, 2015. *The STRIDE Threat Model*, http://msdn.microsoft.com/en-us/library/ee823878%28v=cs.20%29.aspx.

Open Security Foundation (OSF), 2015. *Open Source Vulnerability Database (OSVDB)*. http://osvdb.org.

OWASP Corporation, 2006. *CLASP Comprehensive Lightweight Application Security Process*.

Rolland, C., Prakash, N., Benjamen, A., 1999. A Multi-Model View of Process Modelling, *Requirements Engineering Journal*.

Security Innovation Corporation, 2004. *Hacker Report: Static Analysis Tools*.

Toms, A., 2009. *Threats, Challenges and Emerging Standards in Web Services Security*. Technical report HS-IKI-TR-04-001, Department of Computer Science, University of Skövde.