Computer Standards & Interfaces 36 (2014) 711–722

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Secure Tropos framework for software product lines requirements engineering

Daniel Mellado [a,*], Haralambos Mouratidis [b], Eduardo Fernández-Medina [c]

[a] Spanish Tax Agency, Large Taxpayers Department, IT Auditing Unit, Paseo de la Castellana 106, 28046 Madrid, Spain
[b] School of Architecture, Computing and Engineering, University of East London, 4-6 University Way, Docklands, E16 2RD London, UK
[c] GSyA Research Group, University of Castilla-La Mancha, Information Systems and Technologies Department, Paseo de la Universidad 4, 13071 Ciudad Real, Spain

ARTICLE INFO

Available online 31 December 2013

Keywords:
Security requirements
Product lines
Requirements engineering
Security requirement engineering
Secure Tropos

ABSTRACT

Security and requirements engineering are two of the most important factors of success in the development of a software product line (SPL). Goal-driven security requirements engineering approaches, such as Secure Tropos, have been proposed as a suitable paradigm for elicitation of security requirements and their analysis on both a social and a technical dimension. Nevertheless, goal-driven security requirements engineering methodologies are not appropriately tailored to the specific demands of SPL, while on the other hand specific proposals of SPL engineering have traditionally ignored security requirements. This paper presents work that fills this gap by proposing "SecureTropos-SPL" framework.

1. Introduction

Information systems undoubtedly play an important role in today's society and more and more are at the heart of critical infrastructures. It is widely accepted in the security research literature [14], that security is of particular importance to such information systems and that is essential for security to be considered from the early stages of software development for an effective management of security issues. Although security is traditionally considered a technical issue; security is, in fact, a two-dimensional problem, which involves technical as well as social challenges [18].

At the same time, in recent years, many public and private organizations are making the strategic decision to adopt a software product line (SPL) approach to the production of software-intensive systems [13]. Since SPL strategy has proven successful at reducing both time-to-market and development costs [4,6] and obtaining both high-quality information systems and higher productivity [13]. The SPL development paradigm is based on increasing the reuse of all types of artefacts, thanks to the combination of coarse-grained components with a top-down systematic approach in which software components are integrated into a high-level structure.

Proper analysis and understanding of security requirements are important because they help us to discover any security or requirement defects or mistakes in the early stages of development, in fact the long-standing credo of requirements engineering reads: "If you don't know what you want, it's hard to do it right" [7]. In SPL development it is even more important given that a weakness in security owing to a mistake in a security requirement can cause problems throughout the products of a product line. Therefore, the elicitation of security requirements for SPL is a challenging task, mainly due to the varying security properties required in different products, for the diversity of market segments, and the constraint of simultaneously maintaining the cost-effective principle of the SPL paradigm.

Nevertheless, there is lack of approaches in the security requirements literature [14], which would support the elicitation and analysis of both social and technical security requirements from the early stages of the SPL development process. On one hand current SPL approaches which include partial support for security requirements engineering do not manage both dimensions of security (social and technical dimension); on the other hand, proposals that manage both the technical and the social dimensions of security (such as Secure Tropos) are not tailored enough to support the SPL development paradigm.

In this paper, we propose *SecureTropos-SPL*, an extension of some stages of Secure Tropos [17] methodology to fill this gap. Our work initially aligns SPL concepts to Secure Tropos concepts, and secondly it redefines the Secure Tropos process, so that we proposed a risk-driven goal-based process to manage security requirements variability at both Early Requirements and Late Requirements stages of Secure Tropos in SPL development. Finally, it is proposed the extension of Secure Tropos metamodel and language to support security risks and SPL concepts such as 'variability' and its modelling, that is SPL modelling with Secure Tropos, in order to manage at the same time both the technical and the social dimensions of SPL security and also taking into account the security risks.

This paper is structured as follows. Section 2 describes the background information about Secure Tropos and SPL needed for a better understanding of the proposal. In Section 3 the related work is summed up. Section 4 outlines the core elements of *SecureTropos-SPL*, our

* Corresponding author.
E-mail addresses: damefe@esdebian.org (D. Mellado), H.Mouratidis@brighton.ac.uk (H. Mouratidis), Eduardo.FdezMedina@uclm.es (E. Fernández-Medina).

0920-5489/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.csi.2013.12.006

proposed extensions to Secure Tropos, while Section 5 illustrates with the aid of an example the applicability of these extensions to Secure Tropos. Finally, Section 6 discusses contributions and future work.

2. Secure Tropos and software product lines requirements engineering basics

2.1. Overview of Secure Tropos

Secure Tropos [17] is a security-oriented extension of the widely known requirements engineering methodology Tropos [5]. It introduces a number of security-related concepts to the Tropos methodology. Tropos (and as a result Secure Tropos) methodology is mainly based on four stages:

- *Early requirements* analysis aimed at defining and understanding a problem by studying its existing organizational setting.
- *Late Requirements* analysis conceived to define the system-to-be in the context of its operational environment.
- *Architectural design*, that deals with the definition of the system global architecture in terms of subsystems; and the
- *Detailed design* phase, aimed at specifying each architectural component in further detail, in terms of inputs, outputs, control and other relevant information.

The main unique points of the methodology compared to other security oriented software engineering approaches are that

- Social issues of security are analysed during the early requirements stage;
- Security is considered simultaneously with the other requirements of the system-to-be; and
- The methodology supports not only requirements stages but also design stages.

In this paper we will extend Secure Tropos in order to manage security requirements variability at both Early Requirements and Late Requirements stages of Secure Tropos in SPL development.

2.2. Software product lines requirements engineering basics

A software product line (SPL) is a set of software-intensive systems sharing a common, managed set of features [10] which satisfy the specific needs of a particular market segment or mission and which is developed from a common set of core assets in a prescribed way [6]. Exploiting commonalities between different systems is at the heart of Software Product Line Engineering. These commonalities and differences are described by using the core concept in Software Product Line Engineering: variability. Variability describes the variations in both functional and non-functional features in the product line. Features are either a commonality or a variation. Variability management is the activity in product line development that aims to model a product line as a whole and to customise or change specific product line members. Its importance signifies that it can actually be seen as the key feature that distinguishes product line development from other approaches to software development [23]. In common language use the term variability refers to the ability or the tendency to change, but in this case this change does not occur by chance but is brought about deliberately. For example: an electric bulb can be lit or unlit, or a software application can support different languages. Variability in SPL is therefore variability that is modelled to enable the development of customised applications by reusing predefined, adjustable artefacts. The variability of a SPL thus distinguishes different applications of the product line. In contrast to variability, the commonality in SPL denotes features that are part of each application in exactly the same form. This means that it is often possible to decide whether a feature is a variable of the SPL or whether it is common to all software product applications, and thus adds to the commonality.

The software product line engineering paradigm differentiates two processes: domain engineering and application engineering [21]. Domain engineering is the process of SPL engineering in which commonality and variability of the product line are defined and carried out. According to [21] the domain requirements engineering sub-process encompasses all activities for eliciting and documenting the common and variable requirements of the product line. Application engineering is the process of SPL engineering in which the applications of the product line are built by reusing domain artefacts and exploiting product line variability. Product line requirements define the products and their common and variable features in the product line. Requirements that are common to the entire family, which constitute the product line requirements and an important core asset, should be managed separately from requirements that are particular to a subset of the products (or to a single product), which must also be managed. The SPL scope binds the products included in the product line: product line requirements refine the scope by more precisely defining the characteristics of the products in the product line. Both concepts are closely coupled and evolve together [6].

3. Related work

Several attempts have also recently been made to define SPL architectures for security, such as the approach of Faegri et al. [8] and the approach of Arciniegas et al. [1], although their work is focused on tackling security management in SPL engineering, their approach is applicable to the latest stages of the development process rather than security requirements, because are more orientated towards the software solution than to security requirements elicitation and definition or include only a few security requirements tasks, but without managing all the security requirements artefacts (assets, threats, etc.). The Security Requirements Engineering Process for Software Product Lines (SREPPLine) [15] has been recently proposed to support security requirements analysis for SPL. However, SREPPLine fails to consider both the social and technical dimensions of security and it also does not support the parallel modelling of security requirements and the rest of the security elements and their variability with a homogeneous modelling language, as our approach does.

The most relevant "generic" security requirements related proposals were systematically reviewed in [14] (Secure Tropos included). Thanks to this review that we have already done, it can be observed that these proposals neither are sufficiently specific nor are they tailored to the SPL development paradigm, principally because they do not deal with security requirements variability, which is an essential aspect. Moreover, they do not provide a methodological tailored approach for SPL engineering, that is, they do not have specific activities or language to manage the security variability needed by the SPL development paradigm. Therefore, they are not appropriate enough to manage security requirements in SPL, as it was also explained in [15].

Having said this, each of these approaches makes highly important contributions to security requirements engineering in SPL. In addition, some of their features are used as the basis of our proposal.

4. SecureTropos-SPL: Secure Tropos framework for software product lines

In this section, we present the major principles of our proposal. Firstly, we outline the core of our approach. Next we align SPL concepts to Secure Tropos concepts, and then it is redefined the Secure Tropos process at both Early Requirements and Late Requirements phases of Secure Tropos. Finally, it is proposed an extension of Secure Tropos language in order to deal with the variability needed for SPL engineering and to manage security risks elements.