Towards the Extension of Secure Tropos Language to Support Software Product Lines Development

Daniel Mellado[1] and Haralambos Mouratidis[2]

[1] Spanish Tax Agency, Large Taxpayers Department - IT Audit Unit,
Paseo de la Castellana 106, 28046 Madrid, Spain

[2] School of Architecture, Computing and Engineering, University of East London,
4-6 University Way, Docklands, E16 2RD, London, U.K.

Abstract. The elicitation of security requirements for Software Product Lines (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. Goal-driven security requirements engineering approaches, such as Secure Tropos, have been proposed in the literature as a suitable paradigm for elicitation of security requirements and their analysis on both a social and a technical dimension. Nevertheless, on one hand, 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 an extension to the Secure Tropos language to support SPL.

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 that security is of particular importance to such information systems. Although security is traditionally considered a technical issue; security is, in fact, a two-dimensional problem, which involves technical as well as social challenges [10]. It is also widely accepted, in the security research literature [7], that is essential for security to be considered from the early stages of software development for an effective management of security issues.

Software Product Line (SPL) strategy has proven successful at reducing both time-to-market and development costs [2, 4] and obtaining both high-quality information systems and higher productivity [6].

Nevertheless, there is lack of approaches, 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; on the other hand, proposals that manage both the technical and the social

Mellado D. and Mouratidis H..
Towards the Extension of Secure Tropos Language to Support Software Product Lines Development.
DOI: 10.5220/0004094500690076
In *Proceedings of the 9th International Workshop on Security in Information Systems* (WOSIS-2012), pages 69-76
ISBN: 978-989-8565-15-0

dimensions of security are not tailored enough to support the SPL development paradigm.

In this paper, we propose an extension of the Secure Tropos [9] modeling language to fill this gap. Our work initially aligns SPL concepts to Secure Tropos concepts, and secondly proposes the extension of Secure Tropos language to support 'variability' modeling and therefore SPL modeling.

This paper is structured as follows. Section 2 outlines the core elements of our proposed extensions to the Secure Tropos language, while Section 3 illustrates with the aid of an example the use of the extended language. Section 4 discusses contributions and future work.

2 Overview of the Extensions to Secure Tropos Language to Support Product Lines Development

2.1 Aligning Product Line Concepts with Secure Tropos

Secure Tropos [9] is a security-oriented extension of the widely known requirements engineering methodology Tropos [3]. The main unique points of the methodology compared to other security oriented software engineering approaches are that (i) social issues of security are analyzed during the early requirements stage; (ii) security is considered simultaneously with the other requirements of the system-to-be; (iii) and the methodology supports not only requirements stages but also design stages. Tropos (and as a result Secure Tropos) methodology is mainly based on four phases: early and late requirements, architectural and detailed design. *Early requirements* analysis aims at defining and understanding a problem by studying its existing organizational setting. *Late requirements* analysis defines the system-to-be in the context of its operational environment.

Secure Tropos is a goal driven security requirements engineering methodology, in which a *goal* represents actors' strategic interests and a *secure goal* represents the strategic interests of an actor with respect to security. Secure goals are mainly introduced to achieve possible security constraints that are imposed to an actor or exist in the system. An *actor* is defined as an entity that has strategic goal. In Secure Tropos *security constraints* define the system's security requirements; they are security conditions imposed to an actor that restricts achievement of an actor's goals, execution of plans or availability of resources. In addition, Secure Tropos defines secure dependencies. A *secure dependency* introduces security constraint(s) that must be fulfilled for the dependency to be satisfied.

One of the first challenges we faced, was the introduction of the concept of variability in Secure Tropos, due to the fact that variability management is at the heart of the SPL paradigm.

In SPL, since a goal could provide the rationale for variations in domain requirements [5], we used it as a discriminator that enables us to identify common and variant goals and secure goals in Secure Tropos. Hence, the common (default), optional and/or alternative goals in SPL can be modeled in Secure Tropos by means of a *Variability Dependency relationship* (a new relationship of Secure Tropos explained in Section 3.2 and shown in Fig. 1).

Moreover, in Secure Tropos, the precise definition of how a secure goal can be achieved is given by a *secure plan*, which is defined as a particular way for satisfying a secure goal. Usually, a secure plan or goal needs a *secure resource*, which is an informational entity that is needed for the achievement of a secure goal or the fulfilment of a secure plan. Therefore, these entities of Secure Tropos (security constraint, secure plan, secure resource) could be variants of a SPL because they are related to goals and secure goals which are variations of a SPL, so that they are modeled by means of a variability dependency relationship between them and an actor.

We have also had to adapt the concept of actor of Secure Tropos, so that during Application Engineering the different products/applications instantiated from the SPL are modeled as actors that inherit from the SPL-actor.

A second challenge was to integrate the two main activities related to requirements engineering in SPL engineering with Secure Tropos Process. According to the definitions of these activities, and taking into account the development stages of Secure Tropos, we have integrated the *Domain Requirements Engineering* activity in both the Early and Late requirements phases, and the *Application Requirements Engineering* activity only in the Late requirements phase. That is, for the development of a SPL during the Domain Requirements Engineering activity we will carry out Early and Late requirements phases analyses, initially by defining and understanding the SPL settings and then by defining the SPL-to-be in the context of its operational environment (modeling common, alternative and optional entities). While for the instantiation of the products/applications of the SPL during the Application Requirements Engineering activity we will inherit the early requirements from the SPL completely. Thus, during Application Engineering it will only be needed to carry out the Late Requirements Engineering activity, because is in this activity that each instantiated product/application from the SPL is defined, in the context of its operational environment.

Therefore, through the above discussed alignments and adaptations of concepts as well as the extensions of the two of the Secure Tropos diagrams, i.e. Security Enhanced Actor Diagram (SEAD) and Security Enhanced Goal Diagram (SEGD) explained in next section, we are able to capture and model security, with Secure Tropos, the requirements of a SPL along with the variability of their related entities.

2.2 Abstract Syntax Extension

The abstract syntax of Secure Tropos consists of the Secure Tropos metamodel. In this work we are interested in two sub-parts of the modetamodel related to the Security Enhanced Actor Model (SEAM) and Security Enhanced Goal Model (SEGM).

The SEAM defines a set of actors along with their secure dependencies and any security constraints that might be imposed to these actors. The SEGM assists to analyse the security issues of a particular Actor by understanding the implications that Security Constraints, identified in SEAM, have in that particular actor.

The extension to SEAM is shown in Fig. 1. We have added the '*Variability Dependency'* relationship, which inherits from '*Dependency'* and from which '*Secure Dependency*' inherits, so that variability of *Dependum* entities could be modelled.

Furthermore, through the attribute ‘Depender’ or ‘Dependee’, developers can specify the “owner” of the variant. Secure Dependency relationships are ‘Common’ variants by default.

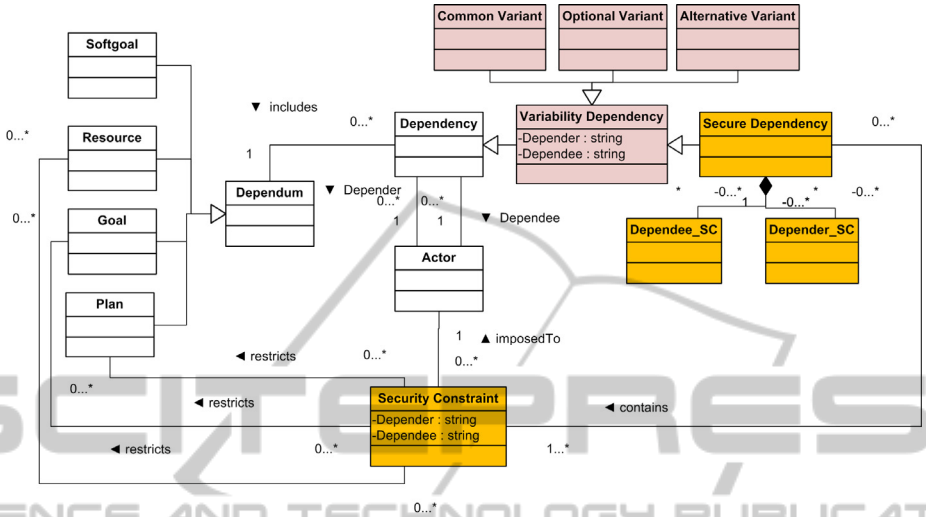

Fig. 1. Extension to SEAM abstract syntax.

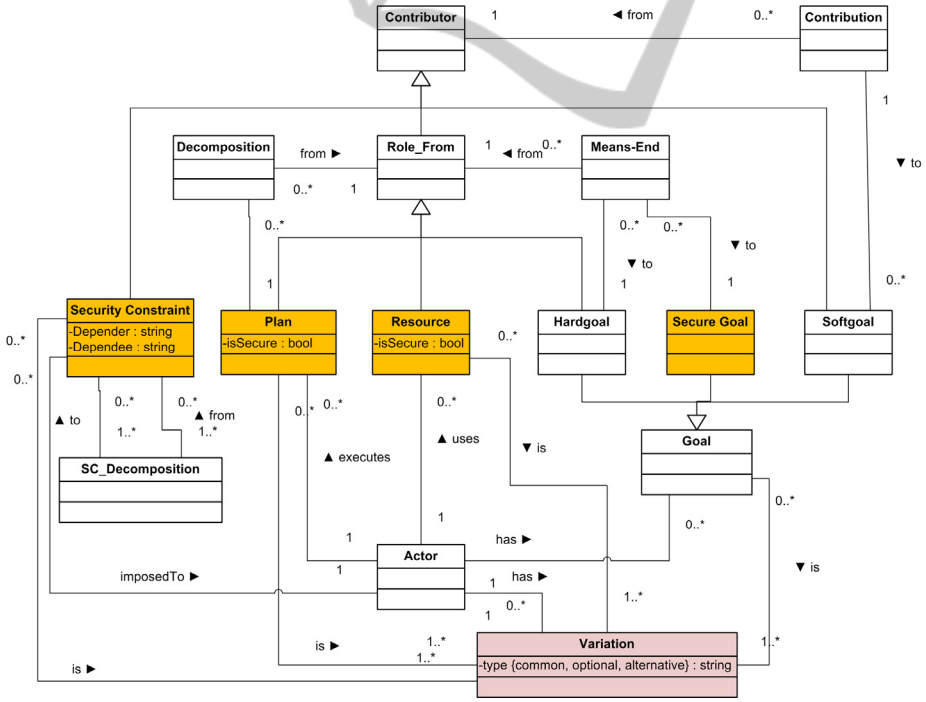

Fig. 2. Extension to SEGM abstract syntax.

In order to manage the variability of the SPL and the instantiated applications from the SPL at the level of a particular Actor, we extended the SEGM. The extension shown in Fig. 2 consists of adding an entity named '*Variation'* which could have as value: 'common', 'optional' or 'alternative', and which is related to the entities *Goal*, *Security Constraint*, *Plan* and *Resource* by means of a relationship '*is part of*' and which an *Actor* could have several *Variation*.

3 Example of Modelling

A simple and short example related to health and social care SPL, is outlined in this section in order to describe and show through the example the extended language and relevant notation of our work.

Graphically, as shown in Fig. 3, Fig. 4 and Fig. 5, in the SEAD (Security Enhanced Actor Diagram) and SEGD (Security Enhanced Goal Diagram) the '*Variability Dependencies*' are represented with '◄*V*' over the dependency that joins the entities, so that the tip of the triangle indicates the "owner" of the variant, i.e. '*Depender*' or '*Dependee*'. In addition, if an entity is a '*Variation*', it is depicted with a '*(V)*' within the representation of the entity. The Secure Tropos entities are represented in the figures as follow: an actor with a circle; a goal with rounded rectangle; a security constraint with an octagon; a plan with a hexagon; and a resource with a rectangle.

This example is based on a real case study described in [8] and it illustrates the application of the extended Secure Tropos language to model the security requirements of a software product line of a CRM (Customer Relationship Management) system, which may have several different configurations for three different public institutions of the public social security system of Spain. Such system, named eCRM, is characterised as a SPL whose members vary by system configuration yet retain the same core functionalities.

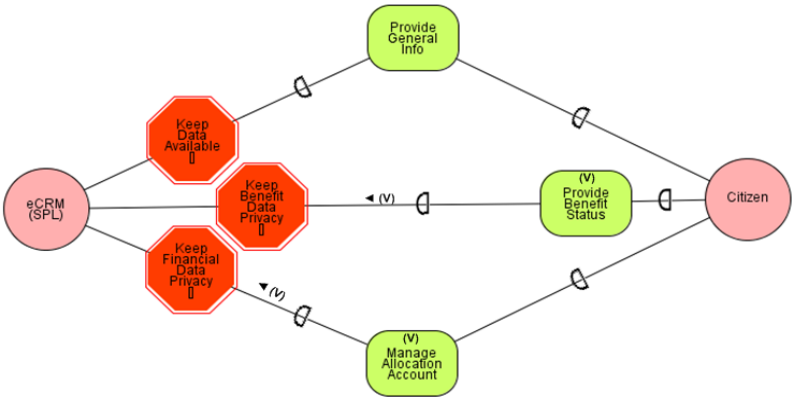

Fig. 3. Part of the SEAD of eCRM (SPL) – (Late Requirements phase).

Fig. 3 shows a SEAD at the Late Requirements phase, which identifies and analyses the actors of the SPL and its environment. It also models the SPL's business

goals, at business and service level, as well as it illustrates the analysis of the variability dependencies of these goals. This means that it supports the modeling of the variability of the goals, specifying the variant goals as common, optional or alternative. As shown in Fig. 3, the *actor* 'eCRM (SPL)' has strategic goals and intentions. In this example, the 'eCRM (SPL)' has a common service goal to citizens: "Provide general information about social security issues" and two optional service goals: "Provide the status of a citizen's benefit" and/or "Manage the allocation account contribution to the Social Security". In order to deal with the security issues, security constrains are introduced along with the variability dependencies. Security constrains, such as those shown in the model ("Keep data available", "Keep financial data privacy" and "Keep benefit data privacy"), represent restrictions related to security that the SPL must have and instantiated products must respect.

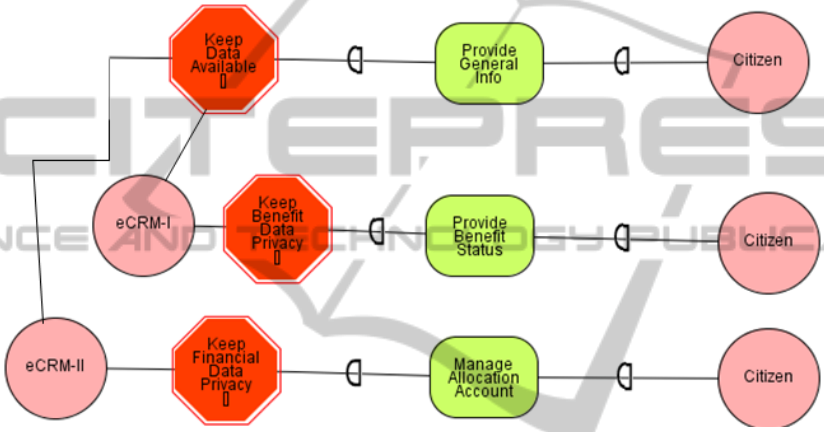

Fig. 4. Part of the SEAD of the instantiated applications of the eCRM (SPL) – (Late Requirements phase).

Fig. 4 illustrates a SEAD at the Late Requirements phase, which models the instantiation of applications from a SPL. In particular, the model represents two applications (eCRM-I and eCRM-II) instantiated from 'eCRM (SPL)', both of which inherit the common goals, constraints, plans and resources. Each application inherits the common business goals from the SPL and the stakeholders of each application choose the optional business goals by exploiting the variability of the eCRM(SPL).

Furthermore, the SEGD shown in Fig. 5 allows a deeper understanding of how the SPL reason about goals to be fulfilled, plans to be performed and availability of resources. It completes the SEAD with the reasoning that each actor makes about its internal goals and constraints, plans and resources. It can be seen that each variant business goal, which is restricted by a constraint, has related secure goals, which satisfy the constraint by means of secure plans that need resources.

Finally, through the entity '*Variation*' of the SEGM it is possible to trace the entities which are part of an instantiated application from the SPL. Hence, by means of this entity of the SEGM we can identify that for the instantiated application eCRM-I, the secure variant entities that are part of it are: the resource 'Benefit database'; the secure goals 'System privacy ensured' and 'User authenticity ensured'; the secure plans related to these secure goals 'Crypto protocol' and 'User authentication'.

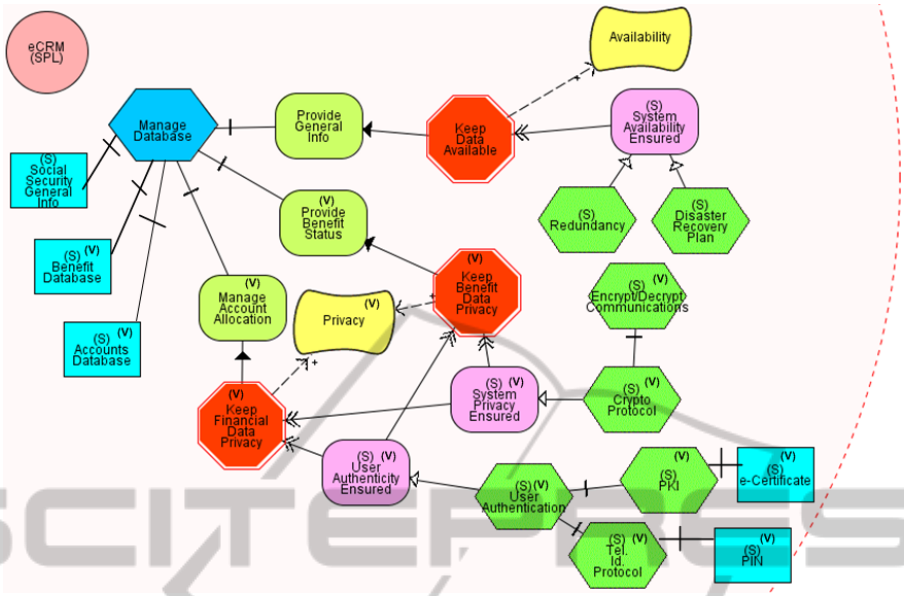

Fig. 5. Part of the SEGD of eCRM(SPL) - (Late Requirements phase).

4 Conclusions and Future Work

A large number of goal-oriented requirements engineering approaches have been proposed in the literature, which focus on eliciting security requirements. However, most of these approaches provide little help as how security requirements can be elicited and modelled in the context of SPL, at both the social and technical dimension, along with the fact that many standard requirements engineering practices must also be appropriately tailored to the specific demands of SPL [1]. This paper introduces the foundations of an approach that fills this gap by proposing an extension to the Secure Tropos language to support SPL. The contribution of this work is that of explaining how SPL concepts are aligned with Secure Tropos concepts as well the presentation of an extension of the language to support 'variability' modeling and therefore SPL modeling.

As future work, we plan to redefine the Secure Tropos process and provide appropriate tool support. This will enable us to apply our work to large and complex case studies and explore its integration with relevant design-level proposals (such as UMLSec in [10]) to facilitate the secure design of SPL.

Acknowledgements

This research is part of the following projects: MEDUSAS (IDI-20090557), financed by the Centre for Industrial Technological Development (CDTI), ORIGIN (IDI-2010043(1-5)) financed by the CDTI and the FEDER, BUSINESS (PET2008-0136)

awarded by the Spanish Ministry for Science and Technology and SEGMENT (HITO-09-138) and SISTEMAS (PII2I09-0150-3135) financed by the Council of Education and Science of the Castilla-La Mancha Regional Government.

References

1. A. Birk and G. Heller, Challenges for requirements engineering and management in software product line development. International Conference on Requirements Engineering (REFSQ 2007), 2007: p. 300-305.
2. J. Bosh, Design & Use of Software Architectures. 2000: Pearson Education Limited.
3. P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini, Tropos: Agent-Oriented Software Development Methodology. 2004: Journal of Autonomous Agents and Multi-Agent System. p. 203-236.
4. P. Clements and L. Northrop, Software Product Lines: Practices and Patterns. SEI Series in Software Engineering. 2002: Addison-Wesley.
5. J. Kim, M. Kim, and S. Park, Goal and scenario bases domain requirements analysis environment, in The Journal of Systems and Software. 2005. p. 926 - 938.
6. J.D. McGregor, Testing a Software Product Line, in Testing Techniques in Software Engineering, P. Borba, et al., Editors. 2010, Springer. p. 104-140.
7. D. Mellado, C. Blanco, L.E. Sanchez, and E. Fernández-Medina, A Systematic Review of Security Requirements Engineering. Computers Standards & Interfaces 2010. 32: p. 153-165.
8. D. Mellado, E. Fernández-Medina, and M. Piattini, Security requirements engineering framework for software product lines. Information and Software Technology, 2010. 52: p. 1094-1117.
9. H. Mouratidis, Secure Tropos: An Agent Oriented Software Engineering Methodology for the Development of Health and Social Care Information Systems. International Journal of Computer Science and Security, 2009. 3(3): p. 241-271.
10. H. Mouratidis and J. Jürjens, From goal-driven security requirements engineering to secure design. International Journal of Intelligent Systems, 2010. 25(8): p. 813-840.