


Chapter 12


Towards a Conceptual Framework for Security Requirements Work in Agile Software Development

Inger Anne Tøndel

 <https://orcid.org/0000-0001-7599-0342>

*Department of Computer Science, Norwegian University of Science and Technology (NTNU),
Trondheim, Norway & SINTEF Digital, Trondheim, Norway*

Martin Gilje Jaatun

 <https://orcid.org/0000-0001-7127-6694>

SINTEF Digital, Oslo, Norway

ABSTRACT

Security requirement work plays a key role in achieving cost-effective and adequate security in a software development project. Knowledge about software companies' experiences of security requirement work is important in order to bridge the observed gap between software security practices and security risks in many projects today. Particularly, such knowledge can help researchers improve on available practices and recommendations. This article uses the results of published empirical studies on security requirement work to create a conceptual framework that shows key concepts related to work context, this work itself and the effects of this work. The resulting framework points to the following research challenges: 1) Identifying and understanding factors important for the effect of security requirements work; 2) Understanding what is the importance of the chosen requirements approach itself, and; 3) Properly taking into account contextual factors, especially factors related to individuals and interactions, in planning and analysis of empirical studies on security requirements work.

DOI: 10.4018/978-1-6684-3702-5.ch012

INTRODUCTION

In today's interconnected world, we would claim that software security is an aspect to consider in most software development projects. Currently, agile development methodologies are prominent in software development. Such methods are used even for large scale development (Dikert, Paasivaara, & Lassenius, 2016) and for security critical and safety critical software (Hanssen, Stålhane, & Myklebust, 2018; Heeager & Nielsen, 2018; Oueslati, Rahman, & ben Othmane, 2015). Thus, good ways of working with security within an agile development paradigm is necessary.

There has been done a lot of work on suggesting ways to deal with software security in agile development projects, including proposals for integrating security into agile methodologies like XP (Aydal, Paige, Chivers, & Brooke, 2006) and Scrum (Pohl & Hof, 2015). In 2009 the Microsoft Security Development Lifecycle (SDL) (Howard & Lipner, 2006; Microsoft, n.d.) was released in a version specific for agile development (Agile SDL) (Microsoft, 2009), and abuser stories have for some time been a suggested way of representing security requirements within agile development (Peeters, 2005). Additionally, there exist method-agnostic approaches to software security that should be possible to integrate with agile development, such as the touchpoints for software security (McGraw, 2004, 2006), the Building Security in Maturity Model (BSIMM) (McGraw, Migués, & West, 2018) and the OWASP Software Assurance Maturity Model (SAMM) (OWASP, n.d.). There thus seems to be no lack of methods for doing software security work also within an agile paradigm. Still, many have observed that security is frequently not given proper attention in software development projects today (Tøndel, Jaatun, Cruzes, & Moe, 2017; Nicolaysen, Sassoon, Line, & Jaatun, 2010; Terpstra, Daneva, & Wang, 2017).

As is well communicated by the abovementioned software security approaches, software security should be an integrated part of development and have a role in the various software development activities, including requirements, design, coding, testing, deployment and operations. Security is not something that can be successfully added on as an afterthought, but should be built into the system from the beginning. This however means that the total number of suggested security activities can be quite overwhelming. It is possible for projects to spend a lot of effort on security, even over-spending, if not properly addressing the security needs. Thus, we consider security requirements work as foundational to achieving cost-effective security in a project.

In this article, we define software security requirements work as activities performed in relation to a software development project to: 1) decide whether and how to identify security needs, risks or requirements for a project; 2) do the requirements elicitation; 3) communicate the identified security needs, risks or requirements, and; 4) integrate these and make priorities related to them in development. By agile development we mean software development that in large part is guided by the Agile principles, as outlined in the Agile Manifesto (Beck et al., 2001), including various methods such as Scrum and XP. Compared to a waterfall development approach, requirements management in agile development is "far more temporal, interactive and just in time" (Leffingwell, 2010). Additionally, the need for requirements prioritization can be considered to be built into the approach; "[w]e admit up front that we can't implement (nor even discover) all potential requirements" (Leffingwell, 2010). Security is only one of the types of requirements a development project needs to consider. When negotiating the three variables cost, schedule and requirements (Leffingwell, 2010), requirements may be modified or dropped altogether.

There exist few empirical studies on how security requirements are handled in software development projects (Terpstra et al., 2017), thus "[h]ow practitioners in the field think about security requirements and how they devise their processes of coping with the issues these requirements pose, is hardly known"

31 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/chapter/towards-a-conceptual-framework-for-security-requirements-work-in-agile-software-development/294467

Related Content

Matilda: A Generic and Tailorable Framework for Direct Model Execution in Model-Driven Software Development

Hiroshi Wada, Junichi Suzuki, Adam Malinowski and Katsuya Oba (2010). *Handbook of Research on Software Engineering and Productivity Technologies: Implications of Globalization* (pp. 250-279).
www.irma-international.org/chapter/matilda-generic-tailorable-framework-direct/37036

Location-Awareness with Action Systems

Luigia Petre, Kaisa Sere and Marina Waldén (2012). *Handbook of Research on Mobile Software Engineering: Design, Implementation, and Emergent Applications* (pp. 463-483).
www.irma-international.org/chapter/location-awareness-action-systems/66483

Eliciting Policy Requirements for Critical National Infrastructure Using the IRIS Framework

Shamal Faily and Ivan Fléchais (2011). *International Journal of Secure Software Engineering* (pp. 1-18).
www.irma-international.org/article/eliciting-policy-requirements-critical-national/61150

Class Imbalance Learning to Heterogeneous Cross-Software Projects Defect Prediction

Rohit Vashisht and Syed Afzal Murtaza Rizvi (2022). *International Journal of Software Innovation* (pp. 1-18).
www.irma-international.org/article/class-imbalance-learning-to-heterogeneous-cross-software-projects-defect-prediction/292021

Prediction of Air Quality Using LSTM Recurrent Neural Network

Supriya Raheja and Sahil Malik (2022). *International Journal of Software Innovation* (pp. 1-16).
www.irma-international.org/article/prediction-of-air-quality-using-lstm-recurrent-neural-network/297982