Agile Threat Assessment and Mitigation: An Approach for Method Selection and Tailoring

Clemens Teichmann, Fraunhofer AISEC, Berlin, Germany
Stephan Renatus, Chef Software Inc., Berlin, Germany
Jörn Eichler, Fraunhofer AISEC, Berlin, Germany

ABSTRACT

Security engineering and agile development are often perceived as a clash of cultures. To address this clash, several approaches have been proposed that allow for agile security engineering. Unfortunately, agile development organizations differ in their actual procedures and environmental properties resulting in varying requirements. The authors propose an approach to compare and select methods for agile security engineering. Furthermore, their approach addresses adaptation or construction of a tailored method taking the existing development culture into account. The authors demonstrate the feasibility of their proposal and report early experiences from its application within a small development organization for digital solutions in the automotive domain.

KEYWORDS

Agile Software Development, Scrum (Software Development), Security Analysis, Threat Modeling

1. INTRODUCTION

Agile development methods are widely accepted and implemented in industry. Often based on the Agile Manifesto (Beck et al., 2001), several proposals for agile methods exist with Scrum as a particularly prominent example (Schwaber & Beedle, 2001).

Security engineering and agile software development are often perceived as a clash of culture. According to Baca & Carlsson (2011), agile practitioners judge security engineering processes as too costly and not beneficial enough in an agile context. Threat assessment and mitigation focus on parts of the product that are either subject to (possibly rapid) change in agile environments – such as code, scope and requirements – or scarcely present at all – such as (architectural) documentation. Corresponding tasks are perceived as documentation-heavy and impeding the fast-moving pace of agile development methods. Requiring a global perspective to become effective (e.g., a system model), these tasks appear to be incompatible with piece-wise product increments that are at the very center of agile methods. In these regards, threat modeling and mitigation is just a part of security risk management, for which Franqueira et al. (2011) compiled a table of mismatches with the agile philosophy.

To address this issue, several approaches have been proposed that allow for agile security engineering (e.g., Jeffries (2012) and Kazerooni & Sethi (2011)). Unfortunately, agile development organizations differ in their actual procedure and environmental properties. Therefore, a specific approach designed for agile threat assessment and mitigation doesn’t necessarily fit to a given agile development organization.
We propose an approach to compare and select methods for agile security engineering focusing on threat assessment and mitigation as prominent example. Our approach applies concepts from the method engineering discipline in order to analyze and disassemble existing methods. Resulting method fragments provide a foundation for the comparison of methods. Utilizing these fragments, a method engineer may adapt or construct a tailored agile threat assessment and mitigation method for an organization taking existing development culture into account.

The remainder of this paper is structured as follows: After this introduction we depict work related to our approach and provide necessary background on method engineering. The following section analyzes and disassembles selected approaches for agile threat assessment and mitigation and presents the resulting method fragments. Section 4 identifies and describes properties that we use to differentiate existing approaches. Application of our approach is demonstrated in section 5 and includes early feedback from a small development organization. A final section concludes and provides an outlook on further research.

2. BACKGROUND AND RELATED WORK

2.1. Method Engineering

Method engineering has been introduced in the 1980s (Bergstra et al., 1985) as a proposal to address diverse requirements with regard to methods needed for the development of information systems. Since the one-size-fits-all approach had been regarded as unattainable, selection and assembly of method fragments in order to provide adequate methods offered a solution. Situational method engineering “encompasses all aspects of creating a development method for a specific situation” (Henderson-Sellers & Ralyté, 2010). As this fits to our problem tailoring a method for a given development environment systematically we apply insights, approaches, and terminology coined for situational method engineering.

According to Henderson-Sellers & Ralyté (2010), a method is a repeatable procedure that specifies the steps involved in solving a specific problem. Standardized building blocks for methods are called method fragments. Within the method engineering discipline, method fragments are commonly classified applying the following dimensions: perspective, abstraction level, and layer of granularity (Brinkkemper, Saeki, & Harmsen, 1998):

- The perspective differentiates product and process fragments: product fragments (“prd” in short) model structures of work products of a method, e.g., diagrams and tables, whereas process fragments (“pro”) are models of the development process, e.g., tasks and activities.
- The abstraction level commonly distinguishes conceptual from technical level. The conceptual level (“conc”) addresses descriptions of development methods whereas the technical level (“tech”) addresses implementable specifications, i.e., tools.
- The compositional aspect is explicated with the layer of granularity: method, stage, model, diagram, and concept. Method fragments on the method layer (“meth”) address a complete method. The stage layer (“stag”) entails method fragments on a segment of the life cycle. The perspective or abstraction of the system to be developed is addressed on the model layer (mod). The diagram layer (“dia”) introduces representations of a view of a model layer method fragment. Concepts and associations of the method fragments on the diagram layer are covered on the concept layer (“conc”).

In general, any method fragment can be classified in all of these dimensions. As an example, Data Flow Diagrams are a product fragment (dimension: perspective) on the conceptual level (dimension: abstraction level) and the diagram layer (dimension: compositional aspect). While method fragments
14 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage: www.igi-global.com/article/agile-threat-assessment-and-mitigation/144787?camid=4v1


Related Content

Analysis of Existing Software Cognitive Complexity Measures  
www.igi-global.com/article/analysis-of-existing-software-cognitive-complexity-measures/204524?camid=4v1a

Online Virtual Learning Environments: A Review of Two Projects  
www.igi-global.com/article/online-virtual-learning-environments/104651?camid=4v1a
A Visual Approach to Business IT Alignment between Business Model and Enterprise Architecture
www.igi-global.com/article/a-visual-approach-to-business-it-alignment-between-business-model-and-enterprise-architecture/123612?camid=4v1a

A Methodology for Adaptive Workflows
www.igi-global.com/chapter/methodology-adaptive-workflows/27843?camid=4v1a