Towards Continuous Security Compliance in Agile Software Development at Scale

Fabiola Moyon
Technische Universität München
Munich, Germany
fabiola.moyon@tum.de

Kristian Beckers
Siemens Corporate Technology
Munich, Germany
kristian.beckers@siemens.com

Sebastian Klepper
Technische Universität München
Munich, Germany
sebastian.klepper@tum.de

Philipp Lachberger
Siemens Corporate Technology
Munich, Germany
philipp.lachberger@siemens.com

Bernd Bruegge
Technische Universität München
Munich, Germany
bruegge@in.tum.de

ABSTRACT

With agile methodologies increasingly being applied in regulated environments, security and compliance emerge as critical issues. Combining both concerns is challenging because security engineering techniques are often based on linear development. We propose a method for achieving continuous and secure development by mapping the requirements of security standards into an agile process model. Additionally, this allows verification of compliance even in the face of dynamic process changes. Applicability of the method is demonstrated by using Business Process Model and Notation (BPMN) to model and extend activities and artifacts of Scaled Agile Framework (SAFe) according to requirements of IEC 62443-4-1, a standard for secure product development in industrial systems.

CCS CONCEPTS

• **Software and its engineering → Agile software development**; • **Security and privacy**;

KEYWORDS

Continuous Software Engineering, Continuous Security, Continuous Compliance, IEC 62443, Scaled Agile Framework, Secure Software Engineering

ACM Reference Format:
Fabiola Moyon, Kristian Beckers, Sebastian Klepper, Philipp Lachberger, and Bernd Bruegge. 2018. Towards Continuous Security Compliance in Agile Software Development at Scale. In *RCoSE'18: RCoSE'18:IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering , May 29, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3194760.3194767

ACM ISBN 978-1-4503-5745-6/18/05…$15.00
https://doi.org/10.1145/3194760.3194767

1 INTRODUCTION

Lean and agile development methodologies are increasingly being used by large enterprises and in industries subject to regulations or security concerns, e.g., financial services, insurance, transportation, manufacturing, health care, or government [17]. Complex organizational structures as well as the need to comply with regulations pose additional challenges [see 13, 7, 9, 15, 11]. Agile principles and secure development practices seem contradictive: Iterative and incremental development complicates risk analysis and assurance activities, focus on functionality neglects security requirements and traceability, dynamic process changes impede audits [4, 1].

The holistic approach of Continuous Software Engineering (CSE) encompasses both concerns and aims to address them continuously but, so far, separately [8]. Our premise is that deep integration into the process is required as opposed to reverting to plan-based activities [cf. 7]. Furthermore, we consider the practice of implementing secure development from a regulations perspective [see 5] as well as the comprehensiveness of standardization (people, process, technology) [see 10] to be excellent sources of requirements. We therefore propose a method to achieve **Continuous Security Compliance** by extending an agile process model to allow secure development as well as verifiable compliance with a standard. We demonstrate this by mapping requirements of IEC 62443-4-1, a security standard for industrial automation and control systems [10], to Scaled Agile Framework (SAFe), a process model for large organizations [14].

2 BACKGROUND

Continuous Software Engineering (CSE) takes a holistic approach to software development across business, development, and operations processes. It utilizes lean and agile principles for rapid and continuous "flow" of activities [8, 12]. Consequently, *Continuous Compliance* seeks to satisfy regulatory requirements on a continuous basis rather than following a "big-bang" approach to ensuring compliance just prior to release of the overall product [8, 9, 15]. *Continuous Security* transforms security from a non-functional requirement to a key concern throughout all phases of the development lifecycle and even post-deployment. This transformation is supported by a smart and lightweight approach to identifying and properly addressing security issues [8, 16].

IEC 62443 is a series of standards for network and system security published by the International Electrotechnical Commission (IEC) with groups focusing on different target audiences in the area of industrial control systems. Group 4 focuses on requirements for component providers for industrial automation and control systems, part 4-1 describes process requirements for secure product development [10].

In addition to team-level activities found in agile process frameworks like Scrum, *Scaled Agile Framework (SAFe)* adds additional levels to cover the needs of large organizations: program, "large solution" (formerly called "value stream"), and portfolio. It incorporates agile and lean practices on all levels and defines corresponding roles, artifacts, and activities [14].

3 RELATED WORK

Bell, Brunton-Spall, Smith, and Bird [6] recognize the need of a bridge between security experts and agile teams. In addition to adapting security tools/techniques with focus on automation, they also discuss challenges for compliance and team security awareness. Our approach fills the gap of a deep analysis on process and people: The extended process model allows agile and security practitioners to review and discuss how well security is covered.

Fitzgerald and Stol [8] analyze several agile software engineering studies and present *Continuous **, their holistic view of CSE. Continuous * describes integrated activities from business, development, operations, and innovation. Our work follows their research agenda regarding Continuous Security as well as Continuous Compliance but combines both concerns and proposes a practical solution.

Baca, Boldt, Carlsson, and Jacobsson [2] present a "security-enhanced" agile development process and evaluate it in the context of financial transactions in a large corporation. In previous works, the authors studied security engineering activities in an agile process and concluded that integrating security is a matter of process flexibility but that "traditional" agile process models are too rigid [3, 1]. We agree with the perspective on flexibility but do not see the need for entirely new process models. Organizations should be able to use any model that suits their needs including, but not limited to, security and compliance.

Fitzgerald, Stol, O'Sullivan, and O'Brien [9] conclude that agile methods are suitable for safety and security critical environments. Nonetheless, they state that "*agile adoption in regulated environments has yet to be addressed*". We aim to specifically fill this gap, providing a step-by-step solution for integrating a security standard with a scaled agile method. Specifically in the security-critical domain: industrial and automation control systems.

Bartsch [4] performs a qualitative study to grasp practitioners' perspectives on secure agile development. In summary, practical concerns are: adapting development process to security, eliciting and tracking security requirements, integrating assurance into iterations. Our approach addresses these concerns in particular and not only promises a solution but also facilitates discussion of experts on both subjects.

4 MODEL-BASED APPROACH

In order to map standard requirements into a process model, both parts need to exist in a common format. We need mergeable models in a notation that can express both sources, particularly roles, activities, and artifacts. We propose using a graphical modeling language to remove the ambiguity of natural language and allow for easier review as well as more focused discussions among experts than a textual description would.[1] Our approach is based on the following structured method to create mergeable models and subsequently combine them to yield an extended process model:[2]

(1) **Modeling: Create separate visual models.** Represent both standards separately using a graphical modeling language as illustrated by figure 1. This way experts can validate models independently and identify mistakes early on. Furthermore, if we directly merged the models at this step, experts might be confused about relationships between models and standards.

[1]In our experience, discussions about standards in natural language often tend to converge around the ambiguity of the text rather than the content and its implications.
[2]Both modeling and merging still involve experts interpreting the standards but visual models enable them to capture their knowledge and collaborate. Subsequently they simplify mapping and keeping track of requirements as well as compliance.

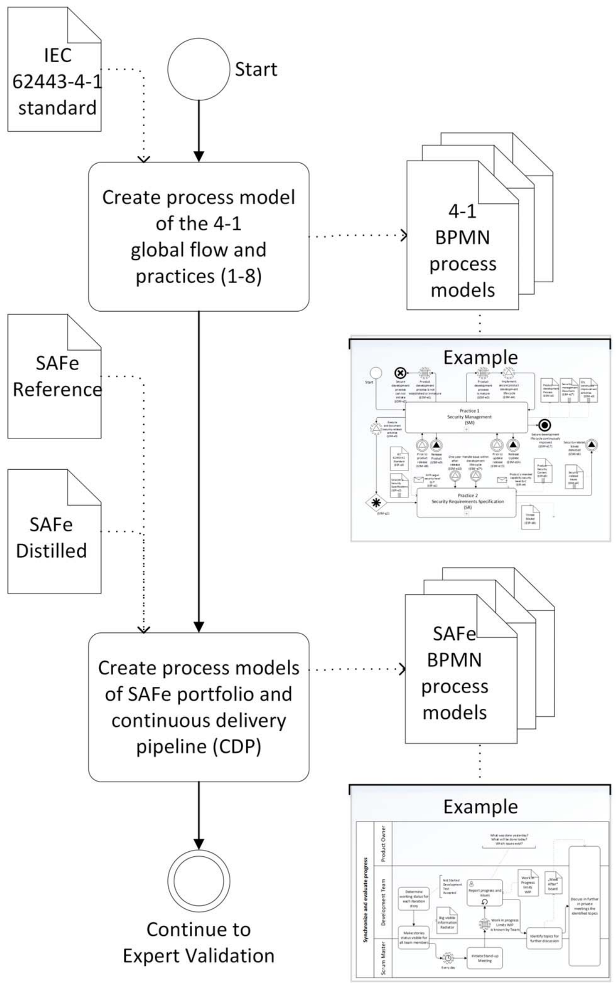
IEC 62443-4-1 standard
Start
Create process model of the 4-1 global flow and practices (1-8)
4-1 BPMN process models
Example
SAFe Reference
SAFe Distilled
Create process models of SAFe portfolio and continuous delivery pipeline (CDP)
SAFe BPMN process models
Example
Continue to Expert Validation

Figure 1: Process of creating separate but mergeable models of standard and process in preparation of expert validation

(2) **Validation: Let practitioners review each model.** Subject-matter experts should validate models in terms of missing, wrong, or misplaced activities as well as connections between activities and artifacts. Ideally, this check is executed with the natural language description and models side by side to verify completeness, correctness, and consistency.
(3) **Merging: Use standard model to extend process model.** Carefully inspect both models to identify similar activities, interfaces, and artifacts. Assign elements of the process model to their corresponding security context, then add or extend elements as necessary to comply with the standard.

As a guideline for assigning process elements to security contexts we suggest software engineering phases. For example, requirements engineering is likely to have commonalities with threat analysis; as a consequence epics, stories, and tasks would be extended with security requirements lists.

5 EXAMPLE

We demonstrate the application of our proposed method using Scaled Agile Framework (SAFe) and IEC 62443-4-1 as practical examples. Our choice of modeling language is Business Process Model and Notation (BPMN) for its simplicity despite its comprehensiveness and expressive power. Figure 2 shows the extended version of one of SAFe's central elements: the *Continuous Delivery Pipeline*. White elements are from SAFe without changes. Gray elements are from SAFe as well but require adaption. Green elements have been added to comply with requirements of IEC 62443-4-1. The process starts with an epic, a high-level description of what the system-to-be shall achieve. Together with the value stream it forms the input of *Prepare and Maintain the Agile Release Train*. This continuous software engineering cycle is repeated throughout the lifecycle of a product. The following notes provide more detail about applying security requirements to process elements:

- For *Build a cross-functional organization* the documentation of roles needs to show that testers of security-critical functionality are independent from developers.
- Security specifications exist for "solutions", which consist of multiple "products". *Define Roadmap* needs to consider those requirements for the particular product roadmap. It also must consider the intended security level and create the *Product Security Context* (a description of the environment the product will be used in), a *Threat Model* of possible attack vectors, and corresponding *Security Requirements*.
- *Plan Program Increment* contains planning and execution of the "PI planning event" which generates several artifacts. IEC 62443-4-1 requires documenting which secure coding standards are used as well as a *Definition of Done* that considers guidelines for secure implementation. These have to become an integral part of the *Standard Development Practices* and *Team Program Increment Objectives*. Security-related issues and problems discussed or recorded during the event must be made explicit and documented separately from other issues in the *Team Program Increment Backlog*.

6 CONCLUSION

We propose a vision for *Continuous Security Compliance* and aim to provide a reference model based on merging visual representations of two well-established standards. We have demonstrated our approach by extending the process model SAFe according to the security standard IEC 62443-4-1 using BPMN notation.

Preliminary results with a subset of both standards show that practitioners find the extended process to be intuitive, precise, and suitable for being applied in their intended environment. Our next goal is to create a precise description of a fully security-compliant version of SAFe. This requires more comprehensive models and validation with experts and practitioners to verify completeness, correctness, and consistency as well as applicability.

The primary use case is guidance on how to comply with IEC 62443-4-1 for agile architects following SAFe. Additionally, we envision easier security certification by compiling required documents from development artifacts as well as checklists for assessing the security maturity of an agile development process.

REFERENCES

[1] Dejan Baca. 2012. *Developing Secure Software - in an Agile Process.* Ph.D. Dissertation. School of Computing, Blekinge Institute of Technology (BTH), Sweden.
[2] Dejan Baca, Martin Boldt, Bengt Carlsson, and Andreas Jacobsson. 2015. A novel security-enhanced agile software development process applied in an industrial setting. In *Proceedings of the 10th International Conference on Availability, Reliability and Security* (ARES '15). IEEE Press, 11–19.
[3] Dejan Baca and Bengt Carlsson. 2011. Agile development with security engineering activities. In *Proceedings of the 2011 International Conference on Software and Systems Process* (ICSSP '11). ACM Press, 149–158.
[4] Steffen Bartsch. 2011. Practitioners' perspectives on security in agile development. In *Proceedings of the 6th International Conference on Availability, Reliability and Security* (ARES '11). IEEE Press, 479–484.
[5] Kristian Beckers. 2015. *Pattern and Security Requirements Engineering-Based Establishment of Security Standards.* Springer International Publishing.
[6] Laura Bell, Michael Brunton-Spall, Rich Smith, and Jim Bird. 2017. *Agile Application Security: Enabling Security in a Continuous Delivery Pipeline.* O'Reilly.
[7] Oisín Cawley, Xiaofeng Wang, and Ita Richardson. 2010. Lean/agile software development methodologies in regulated environments – state of the art. In *Lean Enterprise Software and Systems.* Pekka Abrahamsson and Nilay Oza, (Eds.) Springer Berlin Heidelberg, 31–36.
[8] Brian Fitzgerald and Klaas-Jan Stol. 2015. Continuous software engineering: a roadmap and agenda. *The Journal of Systems and Software*, 123, 176–189.
[9] Brian Fitzgerald, Klaas-Jan Stol, Ryan O'Sullivan, and Donal O'Brien. 2013. Scaling agile methods to regulated environments: an industry case study. In *Proceedings of the 2013 International Conference on Software Engineering* (ICSE '13). IEEE Press, 863–872.
[10] International Electrotechnical Commission (IEC). 2017. 62443-4-1 security for industrial automation and control systems part 4-1 product security development life-cycle requirements. USA, (2017).
[11] Sebastian Klepper, Stephan Krusche, Sebastian Peters, Bernd Bruegge, and Lukas Alperowitz. 2015. Introducing continuous delivery of mobile apps in a corporate environment: a case study. In *2nd International Workshop on Rapid Continuous Software Engineering (RCoSE '15)*, 5–11.
[12] Stephan Krusche and Bernd Bruegge. 2017. CSEPM - a continuous software engineering process metamodel. In *3rd International Workshop on Rapid Continuous Software Engineering (RCoSE '17)*, 2–8.
[13] Dean Leffingwell. 2007. *Scaling Software Agility: Best Practices for Large Enterprises.* Alistair Cockburn and Jim Highsmith, (Eds.) Addison-Wesley.
[14] Dean Leffingwell, Alex Yakyma, Richard Knaster, Drew Jemilo, and Inbar Oren. 2017. *SAFe Reference Guide.* (2017th ed.). Pearson, USA.
[15] Martin McHugh, Fergal McCaffery, Brian Fitzgerald, Klaas-Jan Stol, Valentine Casey, and Garret Coady. 2013. Balancing agility and discipline in a medical device software organisation. In *Software Process Improvement and Capability Determination.* Tanja Woronowicz, Terry Rout, Rory V. O'Connor, and Alec Dorling, (Eds.) Springer, 199–210.
[16] Mark Merkow and Lakshmikanth Raghavan. 2011. An ecosystem for continuously secure application software. *CrossTalk, March/April.*
[17] VersionOne Inc. 2017. 11th Annual State of Agile Report. Tech. rep. Retrieved Jan. 1, 2018 from https://explore.versionone.com/state-of-agile/versionone-11th-annual-state-of-agile-report-2.

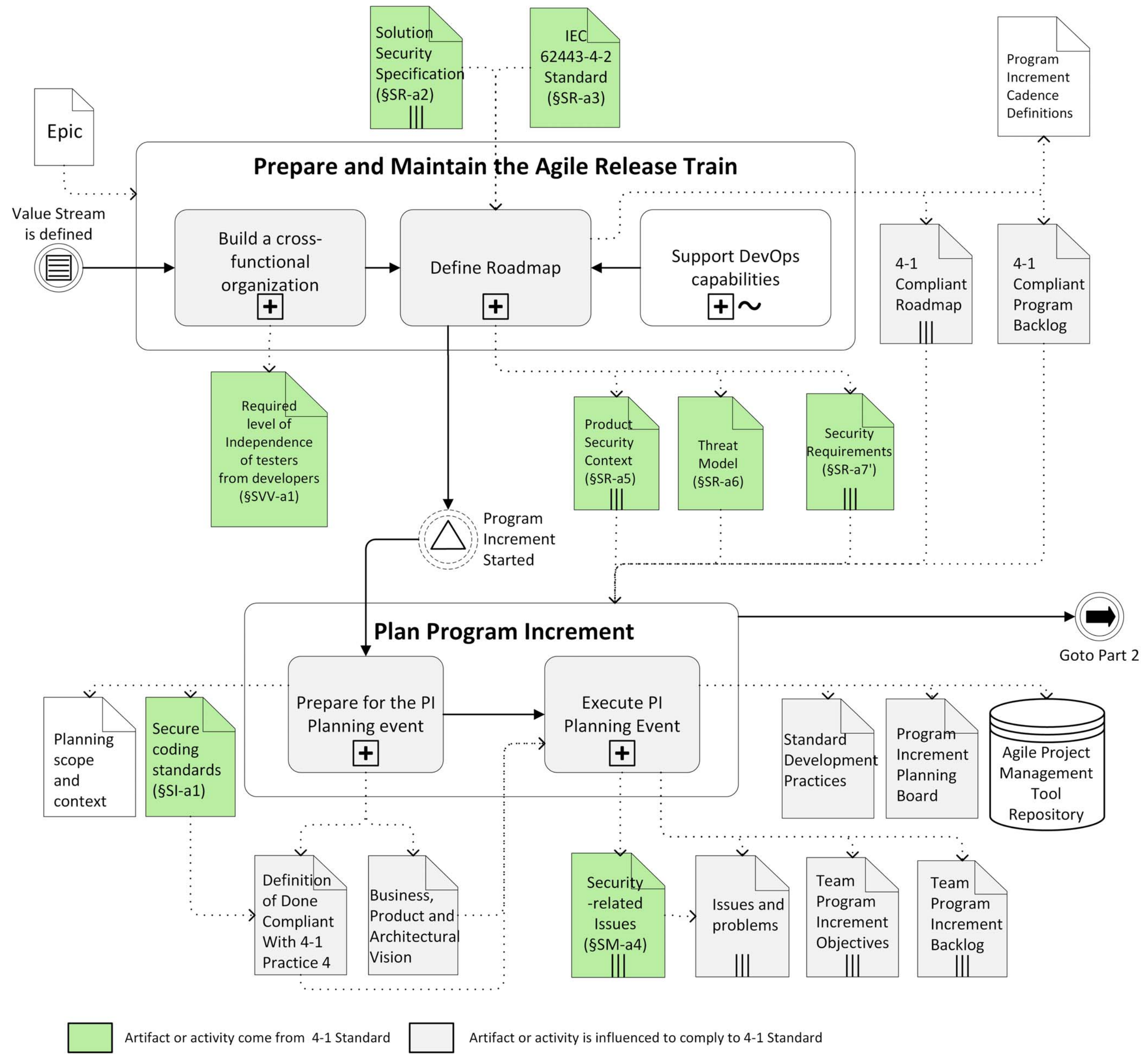
Solution Security Specification (§SR-a2)
IEC 62443-4-2 Standard (§SR-a3)
Program Increment Cadence Definitions
Epic
Prepare and Maintain the Agile Release Train
Value Stream is defined
Build a cross-functional organization
Define Roadmap
Support DevOps capabilities
4-1 Compliant Roadmap
4-1 Compliant Program Backlog
Required level of Independence of testers from developers (§SVV-a1)
Product Security Context (§SR-a5)
Threat Model (§SR-a6)
Security Requirements (§SR-a7')
Program Increment Started
Plan Program Increment
Goto Part 2
Prepare for the PI Planning event
Execute PI Planning Event
Planning scope and context
Secure coding standards (§SI-a1)
Standard Development Practices
Program Increment Planning Board
Agile Project Management Tool Repository
Definition of Done Compliant With 4-1 Practice 4
Business, Product and Architectural Vision
Security-related Issues (§SM-a4)
Issues and problems
Team Program Increment Objectives
Team Program Increment Backlog
Artifact or activity come from 4-1 Standard
Artifact or activity is influenced to comply to 4-1 Standard

Figure 2: Extension of SAFe Continuous Delivery Pipeline to comply with IEC 62443-4-1 practices 2, 4, and 5