Technische Universität München
TUM School of Computation, Information and Technology

Information Security in Large-Scale Agile Software Development

Sascha Nägele

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:		Prof. Dr. Jens Großklags
Prüfende der Dissertation:	1.	Prof. Dr. Florian Matthes
	2.	Prof. Dr. Klaus Pohl

Die Dissertation wurde am 10.07.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 23.01.2025 angenommen.

Abstract

Motivation: The increasing adoption of agile methods at scale, coupled with the escalating importance of security due to rising threats and the substantial financial and reputational damages from security incidents, presents unique challenges for organizations developing software. Academia and industry alike note a significant gap in effective strategies for integrating security within large-scale agile development (LSAD) environments. Key challenges include unclear roles and responsibilities, and the identification of suitable security activities for agile contexts at scale. Particularly critical is the conflict between centralized security governance and the autonomy of development teams. This conflict impacts business and development processes, leading to increased costs, delays, employee dissatisfaction, and clashes between departments. This dissertation offers a comprehensive analysis of this tension from technical, organizational, and human perspectives, providing actionable insights and guidelines to enhance security integration in LSAD and mitigate issues related to autonomy and control.

Research Design: To tackle the identified challenges and research gaps, this dissertation employs a Design Science research strategy, incorporating both primary and supplementary research methods. Primary data collection and analysis methods include systematic literature reviews, expert interviews, a case study, and a survey. Supplementary methods include observations, document analysis, unstructured interviews, and workshops. This mixed-method empirical research approach integrates qualitative and quantitative methods, with a predominant focus on qualitative analysis.

Results: The dissertation, drawing from the results of five core publications, yields several key artifacts. We present a categorization of best practices for integrating security in LSAD across four dimensions and a generic organizational setup outlining the collaboration structure among agile teams with security-related roles at scale. We also provide an overview of 15 challenges categorized into three groups. Furthermore, we evaluate 14 agile security approaches, discussing their benefits and limitations alongside generic evaluation criteria and typical drivers and obstacles for adopting security activities in LSAD environments. In addition, we introduce an adaptive collaboration model for security compliance in LSAD, detailing security-related roles and their activities, and apply this model to prominent large-scale agile frameworks to aid organizational adaptation. We also identify influencing factors and created a scoring model to achieve the adaptivity of our model. Finally, we define criteria to assess the security maturity of agile teams and demonstrate how these criteria could be applied through an exemplary Team Security Maturity Model. In the discussion of our results, we argue that the degree of control and autonomy granted to agile teams should correspond to their capability to develop secure and security-compliant applications, advocating for adjustments based on team maturity levels. This focus on team capability aims to strike a balance between control and autonomy, tailoring security governance to the maturity and needs of each team.

Contribution: Our research provides four key contributions aimed at improving the integration of security in LSAD environments. The first contribution details the current state of challenges, drivers, recurring patterns, and opportunities for integrating security, offering a comprehensive overview of the state-of-the-art. The second contribution delivers practical recommendations and

best practices that encompass roles, responsibilities, security activities, and evaluation criteria. These practices also already suggest methods to mitigate the tension between centralized control and team autonomy. Our third contribution develops these ideas into a collaborative model for effective and adaptable security governance and compliance, synthesizing earlier findings into an operational approach that enhances security integration in LSAD settings. The fourth contribution revolves around guiding the balance between team autonomy and control through the concept of team security maturity. This factor, first identified and detailed in the initial phases of our research, is leveraged within our adaptive approach to address the crucial challenge of balancing control and autonomy in security for LSAD. The dissertation also discusses the further implications for research and practice.

Limitations: Our dissertation addresses the research limitations across our five publications and describes countermeasures to enhance reliability, reproducibility, and validity. We grounded our findings in Systematic Literature Reviews, carefully selecting databases and search terms, applying rigorous inclusion and exclusion criteria, and integrating gray literature to reflect current industry practices. To validate and apply our findings, we conducted structured interview studies with carefully selected participants, recording, transcribing, coding, and analyzing the interviews according to established guidelines. Nonetheless, the practical application of our adaptive and collaborative governance approach requires further research due to the complexity of our research field. A significant overarching challenge was engaging organizations in sensitive security-related research, often resulting in potential contributors being unable to participate.

Future Research: Our dissertation outlines several opportunities for future research derived from our five core publications, along with additional ongoing and future work. Key areas include enhancing the applicability of our adaptive approach and team security maturity model through a comprehensive web application tool support, investigating security metrics, and analyzing the usage of self-assessments. Further research could also explore the creation of a security compliance meta-framework for software development and examine the benefits of agile development for enhancing security.

Zusammenfassung

Motivation: Die wachsende Nutzung agiler Methoden in skalierten Umgebungen, in Verbindung mit der steigenden Bedeutung von Sicherheit aufgrund zunehmender Bedrohungen und der erheblichen finanziellen und rufschädigenden Konsequenzen durch Sicherheitsvorfälle, stellt Unternehmen, die Software entwickeln, vor einzigartige Herausforderungen. Sowohl in der Wissenschaft als auch in der Industrie wird eine erhebliche Lücke bei effektiven Strategien zur Integration von Sicherheit in LSAD-Umgebungen festgestellt. Zu den wichtigsten Herausforderungen gehören unklare Rollen und Verantwortlichkeiten sowie die Identifizierung geeigneter Sicherheitsaktivitäten für skalierte agile Umgebungen. Besonders kritisch ist der Konflikt zwischen zentralisierter Sicherheitssteuerung und der Autonomie der Entwicklungsteams. Dieser Konflikt wirkt sich auf Geschäfts- und Entwicklungsprozesse aus und führt zu erhöhten Kosten, Verzögerungen, Unzufriedenheit der Mitarbeiter und Konflikten zwischen Abteilungen. Diese Dissertation bietet eine umfassende Analyse dieses Spannungsverhältnisses aus technischer, organisatorischer und menschlicher Sicht und liefert umsetzbare Erkenntnisse und Richtlinien zur Verbesserung der Sicherheitsintegration in LSAD und zur Entschärfung von Problemen in Bezug auf Autonomie und Kontrolle.

Forschungsdesign: Um die identifizierten Herausforderungen und Forschungslücken zu bewältigen, wird in dieser Dissertation eine Design Science Forschungsstrategie angewendet, die sowohl primäre als auch ergänzende Forschungsmethoden beinhaltet. Zu den Methoden der primären Datenerhebung und -analyse gehören systematische Literaturrecherchen, Experteninterviews, eine Fallstudie und eine Umfrage. Ergänzende Methoden sind Beobachtungen, Dokumentenanalysen, unstrukturierte Interviews und Workshops. Dieser methodengemischte empirische Forschungsansatz integriert qualitative und quantitative Methoden, wobei der Schwerpunkt auf der qualitativen Analyse liegt.

Ergebnisse: Diese Dissertation, die sich auf die Ergebnisse von fünf Kernpublikationen stützt, liefert mehrere wichtige Artefakte. Zum einen eine Kategorisierung von Best Practices für die Integration von Sicherheit in LSAD über vier Dimensionen hinweg und eine generische Organisationsstruktur, die die Zusammenarbeit zwischen agilen Teams mit sicherheitsrelevanten Rollen in skalierten Umgebungen beschreibt. Zmu anderen einen Überblick über 15 Herausforderungen, die in drei Gruppen unterteilt sind. Darüber hinaus bewerten wir 14 agile Sicherheitsansätze und erörtern ihre Vorteile und Limitationen sowie generische Bewertungskriterien und typische Treiber und Hindernisse für die Einführung von Sicherheitsaktivitäten in LSAD-Umgebungen. Wir stellen ein adaptives Kollaborationsmodell für die Einhaltung von Sicherheitsvorschriften in LSAD vor, in dem wir sicherheitsrelevante Rollen und ihre Aktivitäten detailliert beschreiben, und wenden dieses Modell auf bekannte skalierte agile Frameworks an, um die Umsetzbarkeit unseres Ansatzes zu verbessern. Wir identifizieren auch Einflussfaktoren und entwickeln ein Scoring-Modell, um die hohe Anpassungsfähigkeit unseres Modells zu erzielen. Schließlich definieren wir Kriterien zur Bewertung der Sicherheitsreife agiler Teams und zeigen anhand eines beispielhaften Team-Sicherheitsreifegradmodells, wie diese Kriterien angewendet werden können. In der Diskussion unserer Ergebnisse argumentieren wir, dass das Maß an Kontrolle und Autonomie, welches agilen Teams gewährt wird, ihrer Fähigkeit entsprechen sollte,

sichere und vorgabenkonforme Anwendungen zu entwickeln, und plädieren für Anpassungen der Sicherheitsgovernance auf der Grundlage der Reifegrade der Teams.

Beitrag: Unsere Forschung liefert vier wichtige Beiträge zur Verbesserung der Integration von Sicherheit in skalierte Umgebungen. Der erste Beitrag beschreibt den aktuellen Stand der Herausforderungen, wiederkehrenden Muster und Chancen für die Integration von Sicherheit und bietet einen umfassenden Überblick über den Stand der Technik. Der zweite Beitrag liefert praktische Empfehlungen die Rollen, Verantwortlichkeiten, Sicherheitsaktivitäten und Bewertungskriterien umfassen. In diesen Praktiken werden auch bereits Methoden vorgeschlagen, um das Spannungsverhältnis zwischen zentraler Kontrolle und Teamautonomie zu entschärfen. Unser dritter Beitrag entwickelt diese Ideen zu einem kollaborativen Modell für eine effektive und anpassungsfähige Sicherheitssteuerung und -einhaltung weiter. Damit fassen wir unsere Forschungserkenntnisse zu einem operativen Ansatz zusammen, der das Ziel hat, die Sicherheitsintegration in LSAD-Umgebungen zu verbessern. Der vierte Beitrag dreht sich um die Steuerung des Gleichgewichts zwischen Teamautonomie und Kontrolle durch das Konzept der Team-Sicherheitsreife. Dieser Faktor, der in den ersten Phasen unserer Forschung identifiziert und beschrieben wurde, wird in unserem adaptiven Ansatz genutzt, um die entscheidende Herausforderung des Gleichgewichts zwischen Kontrolle und Autonomie zu bewältigen. In der Dissertation werden auch die weiteren Implikationen für Forschung und Praxis erörtert.

Limitationen: Unsere Dissertation befasst sich mit den Limitationen in unseren fünf Veröffentlichungen und beschreibt Gegenmaßnahmen zur Verbesserung der Zuverlässigkeit, Reproduzierbarkeit und Validität. Wir stützten unsere Ergebnisse auf systematische Literaturrecherchen, wobei wir Datenbanken und Suchbegriffe sorgfältig auswählten, strenge Ein- und Ausschlusskriterien anwandten und graue Literatur einbezogen, um die aktuellsten Praktiken der Branche zu berücksichtigen. Zur Validierung und Anwendung unserer Ergebnisse führten wir strukturierte Interviewstudien mit sorgfältig ausgewählten Teilnehmern durch, wobei wir die Interviews gemäß den festgelegten Richtlinien aufzeichneten, transkribierten, kodierten und analysierten. Die praktische Anwendung unseres adaptiven und kollaborativen Governance-Ansatzes erfordert jedoch aufgrund der Komplexität unseres Forschungsgebiets weitere Erprobung. Eine wichtige übergreifende Herausforderung war die Gewinnung von Organisationen für Forschung in einem sensiblen und sicherheitsrelevanten Themenbereich, was oft dazu führte, dass potenzielle Forschungspartner nicht teilnehmen konnten.

Ausblick: In unserer Dissertation werden mehrere Möglichkeiten für künftige Forschungsarbeiten aufgezeigt, die sich aus unseren fünf Hauptpublikationen ergeben, sowie zusätzliche laufende und künftige Arbeiten. Zu den wichtigsten Bereichen gehören die Verbesserung der Anwendbarkeit unseres adaptiven Ansatzes und des Sicherheitsreifegradmodells durch eine umfassende Unterstützung über eine Webanwendung, die Untersuchung von Sicherheitsmetriken und die Analyse der Nutzung von Selbsteinschätzungsfragebögen. Weitere Forschungsarbeiten könnten sich auch mit der Schaffung eines übergreifenden Frameworks für die Einhaltung von Sicherheitsvorschriften bei der Softwareentwicklung befassen und die Vorteile der agilen Entwicklung für die Verbesserung der Sicherheit untersuchen.

Acknowledgment

I would like to express my deepest gratitude to everyone who has supported me throughout my PhD journey. There are many more people who have positively impacted me than I can list here. Please know that I am thankful for each one of you, even if you are not explicitly mentioned.

First and foremost, I am particularly grateful to my advisor, Prof. Dr. Florian Matthes, who made pursuing my PhD possible in the first place. Your broad and deep expertise, outstanding ability to connect related knowledge, as well as excellent industry contacts, have been invaluable. Alongside your helpful guidance, I greatly appreciate the freedom you allowed me as a PhD student to explore the topics I am most passionate about.

I would also like to thank Prof. Dr. Klaus Pohl for agreeing to be the second advisor of my thesis. I have known your name since I wrote my bachelor thesis on Requirements Engineering in 2013, when your book was an especially helpful resource.

I extend my heartfelt thanks to my (former) colleagues and friends at the chair. I am deeply grateful to have never felt alone on my PhD journey. In particular, I would like to mention, in no specific order: Gonzalo, Nektarios, Tobias, Peter, Uli, Pascal, Alexandra, Stephen, Ömer, Fatih, Martin, Gloria, Jörg, Dominik, Tim, Phillip, Juraj, Oliver, Franziska, Aline and Jian.

I also extend my gratitude to my friends and colleagues from the industry side of my daily activities, including Jenny, Daniel, Tim S., Andreas, Ulrike, Tobias, Corinna, Uwe, Tim L., Rob, Jannis, Jens, and many more.

Working on my PhD on top of my work in the industry, in parallel to teaching and researching at the chair, was not always easy, but thanks to you, it was all worth it.

Special thanks go to my co-authors and students who have worked with me: Nathalie, Philipp, Lorena, Nico, Jann-Lukas, Timo, Markus, Sophie, John, and all my tutors who supported our teaching. It was a pleasure to guide you, and I am grateful for your valuable contributions to our research.

My research was also supported by the SoftwareCampus scholarship, funded by the German Ministry of Education and Research. Thank you to Holger Konle and the entire SWC team for their great support, as well as Stefan, Said, and Danilo from my industry partner DATEV.

I would also like to thank all the industry experts who participated in our research. Without you, this kind of qualitative research based on real industry problems and experience would not have been possible. I am truly grateful for the insights gained from so many organizations.

Finally, from the bottom of my heart, thank you to my family, my partner Hannah, my parents Christian and Elke, my siblings and cousins Lukas, Franzi, Domi, and Alina, my grandparents Ute and Karl, my aunts and uncles, as well as my long-time friends such as Flo, Valerian, Louis, Felix, Sergej, Anna, Kevin, Dennis, Emre, Julian H., Julian D., Robert, Christian and everybody else of our group. Thank you for always believing in me and supporting me in any situation.

Table of Contents

List of Figures

List of Tables

Part A

CHAPTER 1

Introduction

This dissertation focuses on the empirical investigation of the interplay between security and Agile Software Development (ASD) in large-scale environments. Our research systematically identifies challenges and opportunities, and delineates concrete best practices, drawing from both existing literature and industry practice.

The thesis provides a comprehensive perspective that encompasses technical, methodological, and cultural considerations necessary when aiming for a successful integration of security within agile environments with as little friction as possible. Furthermore, it introduces two novel contributions aimed at addressing prevalent issues and bridging identified research gaps, thereby enhancing the efficacy of security within agile environments: a Team Security Maturity Model and an Adaptive Collaboration Model. Additionally, in the outlook, it presents a web application designed to enhance the practical application of these models.

In this introduction, Section 1.1 sets the stage for the dissertation by outlining the motivation behind the topic. Section 1.2 establishes the Research Questions (RQs), followed by Section 1.3 summarizing our research's core contributions. The chapter concludes by presenting the structure of the dissertation in Section 1.4, guiding the reader through the forthcoming chapters.

1.1. Motivation

A critical current challenge in scaled ASD is reconciling the principles of ASD — characterized by user-centricity, iterative development cycles, and autonomous expert teams — with the stringent requirements of IT security and data protection that often necessitate some degree of organizational oversight and control, thereby limiting team autonomy [HBSD18].

This tension is particularly intriguing for two main reasons.

First, the topic's current relevance, which is amplified by the growing adoption of agile methodologies in complex and large-scale projects [SPL+23, vKd23, UPP+22] alongside an increase in the frequency and impact of security incidents. According to the latest "Cost of a Data Breach" report by IBM, the global average cost of a data breach in 2023 was 4.45 million USD, which marks a 15% increase over three years [IBM24]. The data also reveals that 95% of organizations have experienced at least one cybersecurity incident, and 51% of these organizations are likely to increase their investments in security measures [IBM24]. In addition, the "2023/2024 Global Risk Management Survey" by AON [AON] lists cyber attacks and data breaches as the number one risk faced by organizations globally based on the answers of 2842 participating risk, finance, c-suite and HR leaders across 61 countries. In line with the global development, this trend is also observable in Europe. The "ENISA Threat Landscape 2023" report by the European Union Agency for Cybersecurity (ENISA) highlights a significant rise in both the diversity and frequency of cyberattacks, along with their consequences, during the latter part of 2022 and the first half of 2023 [Eur]. Thereby, the topic's relevance even extends beyond the economic and technical perspectives, touching on significant societal challenges, especially as IT- and cybersecurity become crucial in our increasingly digital world. Cybersecurity is seen as one of the key current societal challenges by the European Union, leading to investments through Horizon Europe, the EU's key funding program for research and innovation. The third cluster of Horizon Europe, "Civil security for society", explicitly names cybersecurity as one of the three primary areas of intervention in their work program for 2023-2025 [Com24].

Second, the perceived conflict between agile development practices and security governance introduces multifaceted challenges for organizations developing software, encompassing technical, methodological, and cultural dimensions.

Agile teams, favoring autonomy and rapid iterative value delivery through user-centric software, often encounter friction when their practices intersect with security governance and compliance requirements within large-scale development programs [NWM22, NSM23]. This is particularly evident when regulatory standards, such as IT security laws or data protection regulations, come into play, requiring adherence to specific quality standards and procedures. In addition, self-imposed internal or industry-specific standards to protect a company's reputation, operations, and assets often lead to the use of central governance and compliance teams [NWM22]. For example, a central information security team may impose rigorous testing, review and acceptance procedures, as well as adherence to centrally defined requirements which may be less suitable or even confusing for some individual teams, leading to delays and friction within the development process. For instance, the launch of a newly developed feature eagerly awaited by the business side may be postponed because important security requirements have been neglected or explicit documentation of the compliance is required [MAR+20].

The contrasting approaches and value systems between agile teams and governance roles can be conceptualized through Edward T. Hall's [Hal73] theory of "high-context" and "low-context" cultures [Wri14], highlighting the communication and collaboration disparities that exacerbate this tension. *High-context cultures* tend to be more collectivist, group-harmony oriented, and trust-based, whereas *low-context cultures* prefer a logical, fact-oriented communication with precise language and fixed commitments [Hal73, Wri14]. In terms of the roles involved in this dichotomy, members of agile teams often place more emphasis on contextual aspects, while those

in audit and governance roles typically exhibit characteristics of low-context cultures [Wri14]. These traits are often advantageous for the main responsibilities of these roles, but as shown through the research in this thesis, there is a whole set of challenges resulting from these conflicting characteristics.

At first glance, a seemingly straightforward resolution to the conflict might involve the elimination of either security governance and compliance or agile methodologies within large-scale development environments. However, this strategy poses considerable obstacles and drawbacks. Abandoning central oversight in favor of exclusive reliance on decentralized, autonomous teams typically falls short for three main reasons.

First, agile frameworks and methods such as Scrum [Scr] do not provide any or only very little guidance on how to address security requirements and compliance concerns, which might also even be misplaced in such a small-scale software management framework like Scrum [TP17]. However, even when extending the scope to large-scale agile, our analysis of the most commonly used scaled agile frameworks such as Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS), or Scrum of Scrums demonstrated that these frameworks do not provide enough guidance nor adequately address, embody, or maintain regulatory demands over extended periods, even if some of those frameworks at least recognize the importance of security and other quality requirements [NSM23]. These findings are also backed by previous research, reporting challenges such as unclear roles and responsibilities [vBS18] and causing the development of extensions of these approaches to better address security [SPD21, MMBK21, DCB19].

Second, the autonomous nature of agile teams, while beneficial for certain aspects of development, frequently proves insufficient for meeting stringent security requirements due to varying levels of maturity and team capability in security practices [NWM24] such as risk and threat analysis, security architecture and code reviews, or penetration testing.

Third, in large organizations, depending on the industry and relevant regulatory requirements, development teams often face similar security demands and challenges, which may lead to redundant efforts in decentralized environments [NWM22]. A central security team can increase efficiency by bundling security efforts and streamlining aspects such as providing necessary documentation and evidence of security measures during audits by internal or external auditors, or state supervisory authorities. This, in turn, helps the development teams as they can focus more on their core tasks.

These challenges become especially pronounced in sectors requiring heightened security measures, such as finance, healthcare, or other critical infrastructure [BBCJ15, BSM20].

Nonetheless, forgoing agile development methods is often not advisable as it can diminish market competitiveness [WRH+22] and employer attractiveness [KS21, RZ22]. Even from a security perspective, iterative agile approaches have crucial advantages. For example, they allow for the swift planning and execution of necessary enhancements, contrasting with the extended timelines associated with traditional development cycles.

The objective of this dissertation, therefore, is to harmonize these conflicting perspectives, advocating for a balance between agile teams' autonomy and organizational control to achieve secure and security-compliant software products.

To accomplish this goal, our research aims to thoroughly understand this conflict by examining existing challenges, best practices, and the benefits of integrating security governance in large-scale agile environments, informed by both academic literature and industry practices. For this purpose, within an overarching Design Science (DS) research process, we conducted Systematic Literature Reviews (SLRs), semi-structured expert interview studies, and a case study including document analysis, observations, interviews, and a survey.

Based on these results, we propose two central solution artifacts, in addition to multiple smaller solution artifacts, to balance the autonomy-control tension in the context of security within Large-Scale Agile Development (LSAD).

The first main artifact is an adaptive process tailored to the specific circumstances, roles, and activities of security at scale, allowing for variable levels of control based on influencing factors such as product risk and development team capabilities.

The second main artifact is a team security maturity model designed to evaluate an agile team's ability to develop secure and security-compliant applications, influencing the adaptive process and enhancing security integration in agile development.

This integration is vital for improving software security, thereby safeguarding sensitive data, customer trust, and competitive know-how, while maintaining the benefits of agile development methods at scale.

Thereby, this dissertation addresses the identified challenges and gaps in literature and practice, and provides four key contributions, as detailed in Section 1.3.2. Our research objectives leading to these contributions are described in the next section.

1.2. Research Objectives

This section presents the RQs that the thesis aims to answer.

Research question 1 (RQ1)

What is the current state of information security and security governance and compliance in (large-scale) agile development?

The first RQ aims to build a deep understanding of the interplay between agile development and security by examining the current state of integrating security governance in agile environments in research and practice, with a focus on large-scale environments. When referring to the current state of security in LSAD, more specifically, we address the following important sub-questions:

- What are recurring challenges of security in LSAD?
- What do existing solution approaches and best practices for integrating security (governance and compliance) in LSAD environments look like?
- What security activities are being used in LSAD environments?
- What are the drivers of adopting security approaches in LSAD?

- What obstacles occur when adopting security approaches in LSAD?

By employing a SLR and an expert interview study, we also identified four key areas that are important to analyze the state of security in LSAD:

- The structure of the agile program
- Security governance
- Integration of security activities
- Automation & tool-support

We used these areas to guide our research and structure our studies and results addressing the first RQ, providing detailed insights into the current state of security (governance and compliance) in LSAD. To achieve detailed insights into practice, we complemented our SLRs and interview studies with a case study in the finance industry, including document analysis, observations, interviews, and a survey.

Research question 2 (RQ2)

What are security-related roles, activities, and key influencing factors to balance the autonomy and control tension and improve the security in LSAD integration?

The second RQ aims to deepen and expand the gained understanding from the first RQ. We focus on extracting and combining suitable best practices and solution approaches to improve the identified challenges. Specifically, we propose organizational structures with essential roles and responsibilities to support shifting security responsibility to agile teams in a sustainable way. We consider and leverage the advantages of both the security governance and compliance as well as the agile development perspective. Furthermore, we evaluate and recommend security activities suitable for LSAD environments and propose key influencing factors to balance the autonomy control tension when shifting security responsibility to agile teams on a large scale.

Research question 3 (RQ3)

How can a parameterized adaptive process and collaboration model integrate security in large-scale agile development to achieve both agility and security?

With the third RQ, we proceed one step further and combine the previously identified aspects to balance the autonomy control tension and support the security in LSAD integration by proposing an adaptive process and collaboration model. This model facilitates a systematic balancing of control and autonomy mechanisms and thereby allows more autonomy for capable teams while still aiming for auditability and compliance with security regulations. The model is based on all our previous empirical findings, and evaluated by industry experts with a broad range of relevant roles such as agile developers, product owners, Security Engineers (SEs), governance and compliance experts, and auditors.

Research question 4 (RQ4)

How can a maturity model be designed and implemented to assess a team's capabilities to develop secure and security-compliant applications?

With the fourth and final RQ, we complete our research by exploring in detail one specific influencing factor identified in RQ2 and used in RQ3 within the adaptive process: The maturity or capability of a team to develop secure and security-compliant applications. Based on a SLR and interview study, we propose and evaluate the most important criteria to assess and evaluate the security maturity of an agile development team. In addition, we propose the Team Security Maturity Model (TSMM), an exemplary model derived from the identified criteria. Furthermore, we provide guidelines and advice on how to leverage security maturity on a large-scale.

1.3. Contributions & Publications Summary

This dissertation resulted in **five first-author publications to answer the four RQs of this thesis, providing four main contributions and 41 key findings**.

In the following sections, we summarize these publications, explain their interconnections, and outline how they collectively lead to the key contributions of this dissertation.

In addition, we briefly describe two additional publications that originated during the beginning of our research and built a basis for the shift to the focus of our core publications.

1.3.1. Publications

We provide a short summary of each publication in the following. Part B (Chapter 4) of this thesis lays out the fact sheets of each of our core publications, including each publication's abstract. Appendix A contains the full version of all core publications of this thesis.

Table 1.1 shows an overview of the publications associated with the four dissertation's key contributions.

The rankings of the publication outlets included in the table were accessed on 24.04.2024 using the conference ranking tool ICORE [ICO24], an international collaboration based on the established CORE rankings, for **P1**, **P3**, and **P5**. Since there are no ICORE/CORE rankings for the outlets of **P2** and **P4**, we also included the Qualis Computer Science Conference Rankings [Qua24] for all the publication outlets of **P1-P5**.

P1: In our initial publication [NWM22], "Investigating the Current State of Security in Large-Scale Agile Development", we conduct an empirical study focused on tackling software product security within LSAD environments. Drawing from a literature review and expert interviews across nine companies, we pinpoint four key factors influencing how to handle security in LSAD: (i) the structure of the agile program, (ii) security governance, (iii) adaptions of security activities to fit agile processes, and (iv) tool support and automation. Our analysis also reveals recurring patterns of best practices and challenges in those categories and identifies differences between

the analyzed companies. Hence, **P1** mainly answers RQ1, and contributes to answering RQ2, thereby also building a foundation for RQ3 and RQ4.

P2: Our second publication [NSM23] with the title "The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study" builds upon the foundational overview provided in **P1** and further enriches the captured state-of-the-art by systematically analyzing the current challenges and solution approaches of security governance and compliance integration in LSAD. We present 15 relevant challenges, a LSAD framework analysis, an overview of integration strategies, and five factors designed to balance the autonomy control tension. Thereby, **P2** primarily contributes to answering RQ1 and RQ2, while laying a critical groundwork for RQ3, in particular the influence factors that are further explored and integrated into a comprehensive model in **P4**.

P3: In our third publication [NKM23] titled "Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry", we present the results of an empirical investigation, analyzing the adoption of agile security approaches identified in the literature. We carried out observations, document analysis, and unstructured interviews to identify which approaches the case company applies. We then conducted semi-structured interviews with 10 experts and a survey with 62 participants to evaluate 14 approaches. We also contribute evaluation criteria as well as drivers and obstacles for the adoption of agile security approaches that can be used for further research and practice. Therefore, we mainly contribute to finalizing the answer to RQ1 and RQ2, but also provide interesting insights that are of relevance for RQ3 and RQ4, for example, the selection of suitable security practices for LSAD environments.

P4: Our fourth publication [NSFM24], "Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development", introduces a novel adaptive security governance strategy for LSAD. This approach, grounded in Design Science Research (DSR) and validated through expert interviews, encompasses a standardized organizational structure for security roles, a model to assess team autonomy, and a flexible collaboration framework. Our model delegates specific activities to roles and adjusts their frequency according to the team's autonomy, effectively managing the balance between autonomy and control while maintaining compliance. While our approach is designed to be framework-agnostic, we also describe its effectiveness within established scaling agile frameworks to confirm its broad applicability. **P4** primarily answers our third RQ.

P5: In our fifth publication [NWM24], "Assessing Team Security Maturity in Large-Scale Agile Development", we deepen our investigation of one of the main parameters in our adaptive model from **P4**, which is the security maturity of the development team. The approach presented in **P5** eases the conflict between security governance and development agility through a criteria-based security maturity assessment. This assessment empowers mature agile teams with greater autonomy. Developed using DSR, complemented by a literature review and an interview study, we put forward two principal contributions: a set of criteria for evaluating a team's security capabilities and a team security maturity model. Expert evaluations validate the utility of these tools in systematically measuring team capabilities to produce secure and compliant software. Consequently, organizations can afford providing more autonomy to teams with higher maturity

while focusing resources on enhancing less mature teams. This publication primarily addresses our fourth and final RQ, RQ4.

Table 1.1.: Overview of core (embedded) dissertation publications.

No.	Title	Outlet	Ranking	Pages
C1: *Current state of security in LSAD.* **C2**: *Roles, activities, and key influencing factors to balance the autonomy and control tension.*				
P1	Investigating the Current State of Security in Large-Scale Agile Development	XP	**B/A3**	**17**
P2	The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study	IEEE CBI	**-/A4**	**10**
P3	Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry	ARES	**B/A2**	**12**
C3: *Adaptive process and collaboration model to improve security integration in LSAD.*				
P4	Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development	ICEIS	**-/A3**	**12**
C4: *Maturity model to assess team security maturity.*				
P5	Assessing Team Security Maturity in Large-Scale Agile Development	HICSS	**A/A1**	**10**

Abbreviations: C = Contribution, P = Publication, CON = Conference. Outlet: XP = 23th International Conference on Agile Software Development in Agile Processes in Software Engineering and Extreme Programming, IEEE CBI = 25th IEEE Conference on Business Informatics, ARES = 18th International Conference on Availability, Reliability and Security, ICEIS = 26th International Conference on Enterprise Information Systems, HICSS = 57th Hawaii International Conference on System Sciences. Ranking: The first value in the X/Y value pair represents the ICORE ranking from A (best) to C (worst), the second value represents the Qualis ranking from A1 (best) to B4 (worst).

1.3.2. Contributions

In the following, we describe the main four contributions of our thesis and the individual impact of our five core publications towards these contributions:

Contribution I: Revealing challenges, drivers, opportunities, recurring patterns and solution approaches for security and LSAD integration.

Our first contribution is a detailed look into the current state of security in LSAD environments, which answers our first RQ.

Due to the lack of a comprehensive and systematic analysis of the current state in literature and industry, we conducted a SLR and an interview study, published in **P1**. We have identified four main categories that are crucial for tackling security in LSAD: (i) the structure of the agile program, (ii) security governance, (iii) adaptions of security activities to agile processes, and (iv) tool-support and automation. We propose these categories based on our SLR results and findings from industry experiences extracted from our semi-structured expert interviews. For instance, in the structure of the agile program category, we describe typical roles and responsibilities and

illustrate a generic organizational setup with roles and their interactions for achieving security in LSAD based on our study results. In total, we describe 17 recurring patterns of best practices in the four categories and describe differences and similarities between the analyzed organizations. Thereby, we contribute to the scarce empirical evidence on how to tackle security in LSAD.

Even though our research in **P1** has already provided some intriguing insights into current challenges, the scope, context, and understanding of the current challenges were still relatively limited. We consequently carried out a second study **P2** with a higher focus on challenges and contributed a description of 15 relevant challenges in three categories, based on a more focused SLR and a larger number of interviews: (i) security challenges in LSAD, (ii) selected security challenges in (small-scale) agile development, transferable to LSAD, and (iii) selected general challenges in LSAD, transferable to a security context.

Furthermore, we contribute solution approaches for the security in LSAD integration extracted from the academic literature. These include enhanced LSAD approaches, enhanced small-scale agile approaches as well as the adoption of Secure Software Engineering (SSE) practices.

We also analyzed existing scaling agile frameworks regarding their solution approaches for security integration, resulting in an overview of different frameworks and their maturity in the areas of security, governance, and compliance.

Due to the very limited best practices found in LSAD frameworks, we conducted another study including a SLR for identifying solution approaches described in academic literature, which resulted in a list of 27 agile security approaches. We then evaluated these security approaches in practice by conducting a case study in the finance sector over the period of half a year where we analyzed the current state of security in LSAD based on the case company and published the results in **P3**. The case study included unstructured and semi-structured interviews, document analysis, observation, and a survey. In addition to our core focus on agile security activities, our case study also revealed 11 drivers and obstacles for the adoption of agile security approaches

Contribution II: Presenting best practices for security integration in LSAD (roles, responsibilities, activities, evaluation criteria, and control autonomy balancing factors).

Driven by the large number and impact of the identified challenges in both academic literature and practice, and the simultaneous lack of guidance on how to deal with these hindrances, we also directly collected reoccurring solution approaches in our studies in parallel to identifying challenges, as described in the first contribution.

In our second contribution, on the other hand, we provide best practices that go beyond the recurring patterns observed and derived from literature and empirical evidence. Even if those resulting best practices are of course still rooted in previous results, as explained in detail in the respective publications, we provide more context, recommendations, and guidance based on our own novel research compared to the first contribution that focuses on revealing the state-of-the-art.

Notably, in **P2**, we introduce five key factors designed to harmonize control and autonomy, aiming to alleviate security challenges in LSAD.

Additionally, our third publication, **P3**, presents the evaluation of 14 agile security methodologies through a detailed case study, offering recommendations and guidance on their applicability across different scenarios.

In addition, we contribute evaluation criteria for agile security approaches that can be used by organizations or researchers to score the potential usage and fit of security activities in their environments based on three factors: the alignment with agile methods, suitability for large-scale environments, and impact on regulatory requirements.

Finally, in **P4**, we detail recommendations for organizational structures, including a delineation of roles and responsibilities. **P5**, on the other hand, identifies success factors for agile teams in achieving security within LSAD settings. However, these two publications **P4** and **P5** contribute mainly to the third and fourth contributions of this thesis.

Contribution III: Improving the security in LSAD integration through an adaptive governance approach.

Using the best practices presented in the second contribution, some of the challenges identified in the first contribution can already be overcome.

Nonetheless, unresolved challenges persist, such as the ambiguity surrounding security roles and responsibilities in LSAD and the orchestration of various solution approaches addressing different sub-problems into one unified cohesive strategy. In particular, alleviating one of the most significant issues identified through our research, the autonomy and control conflict, requires such a harmonized approach.

To fill this gap, our fourth publication, **P4**, introduces an adaptive approach for security governance in LSAD. This approach encompasses a flexible organizational structure for security roles, a model for assessing team autonomy, and an adaptive collaboration model.

This model assigns specific activities to designated roles and adjusts their frequency based on the level of team autonomy, thus mitigating the tension between autonomy and control while ensuring regulatory compliance.

Although our model is designed to be independent of any specific agile methodology or framework, we applied it to existing scaled agile frameworks to verify its applicability.

The foundation for this contribution is laid by publications **P1**, **P2**, and **P3**, which identify the core challenges that our solution model in **P4** aims to address. Moreover, **P4** synthesizes promising solution approaches and industry insights from **P1-P3**, consolidating them into a cohesive adaptive strategy.

P5 on the other hand puts the spotlight on one specific factor that is used in the team autonomy assessment model: the team maturity, which describes criteria to measure the capability of a development team to build secure and security-compliant applications, as presented in the next paragraph as the fourth and final contribution.

Contribution IV: Guiding the team autonomy and control balance based on team security maturity.

In our previous contribution, we addressed the autonomy-control conflict and proposed primary

influence factors for its mitigation. Building upon this, **P5** delves specifically into one of those factors: team security maturity, exploring the specific criteria that empower a team to create secure and compliant applications. We introduce ten critical criteria for assessing this capability, derived from academic research through a SLR and further extended, refined, and validated through semi-structured expert interviews.

To illustrate the applicability and demonstrate how these criteria can be utilized to derive a team security maturity score, we present an exemplary "Team Security Maturity Model" (TSMM) that includes potential calculation and assessment approaches. These approaches blend self-evaluations, external assessments, and metrics obtained from semi-automated tools like static and dynamic application testing.

As described in the previous contribution, the results of **P5** are situated within the broader framework of **P4** that enhances its relevance and utility by incorporating it into an adaptive process. This includes considering other crucial factors such as product risk, thereby providing a comprehensive approach to managing the balance between autonomy and control.

1.3.3. Additional Related Publications

In addition to the embedded publications of this dissertation, we achieved two additional, second-author publications [UNH19, UNHM21] (see Table 1.2) related to this thesis, which might interest readers. However, the results of these publications are not part of this dissertation. These two publications marked the starting point of our current research by exploring a specific method of IT governance, namely architecture principles and guidelines, in LSAD.

In these publications, we addressed challenges in LSAD such as inter-team coordination and communication, balancing intentional and emergent architecture, and coordinating development activities to achieve enterprise-wide standards. Using a mixed-method approach, we introduced a solution concept for collaboration between enterprise architects, solution architects, and agile teams. This concept relies on institutional pressures, as presented in our first additional publication [UNH19], and includes a web application tool support [UNHM21], detailed in our second additional publication.

Through our research and industry partner studies, we observed that similar problems exist in the area of security, but, in our impression, to an even greater and more complex extent. It became apparent that these issues cause high costs and dissatisfaction due to conflicts between security and agility at scale, prompting us to shift our research focus to security in LSAD.

Table 1.2.: Overview of additional related publications.

No.	Title	Outlet	Type
AP1	Establishing Architecture Guidelines in Large-Scale Agile Development Through Institutional Pressures: A Single-Case Study	AMCIS	CON
AP2	A Tool Supporting Architecture Principles and Guidelines in Large-Scale Agile Development	Springer	BC

Abbreviations: AP = Additional Publication, CON = Conference, BC = Book Chapter. Outlet: AMCIS = Americas Conference on Information Systems (2019).

1.4. Dissertation Structure

To answer our four RQs and detail the results of our four contributions, we chose a three-part dissertation structure that we will describe in the following.

An overview of the thesis structure is shown in Fig. 1.1.

Part A includes three chapters:

Chapter 1 provides an introduction to the readers of the dissertation by motivating the importance and current relevance of security in LSAD (Section 1.1), highlighting our research objectives (Section 1.2), and summarizing our contributions (Section 1.3).

Chapter 2 contains all the necessary theoretical background to prepare the reader for following the main five included publications of this thesis as well as the results summary and discussion. In particular, we present the foundations of ASD and LSAD in Section 2.1 and the background and definitions regarding information security and related concepts in Section 2.2. Subsequently, we dive into security governance and compliance (Section 2.3), secure software engineering (Section 2.4) and maturity models (Section 2.5). Finally, the interplay between these areas is presented in Section 2.6.

To conclude Part A of this thesis, in Chapter 3, we explain our overarching research strategy (Section 3.1) and applied research methods (Section 3.2).

Part B presents the fact sheets of our five embedded publications.

Part C consists of three chapters:

In Chapter 5, we summarize our results per RQ (Section 5.1), present and discuss the key findings of our research (Section 5.2), outline the implications for research and practice (Section 5.3), and consider the limitations of our research (Section 5.4).

In Chapter 6, we provide an outlook with an overview of ongoing and future work.

Chapter 7 wraps up the dissertation by presenting the conclusion.

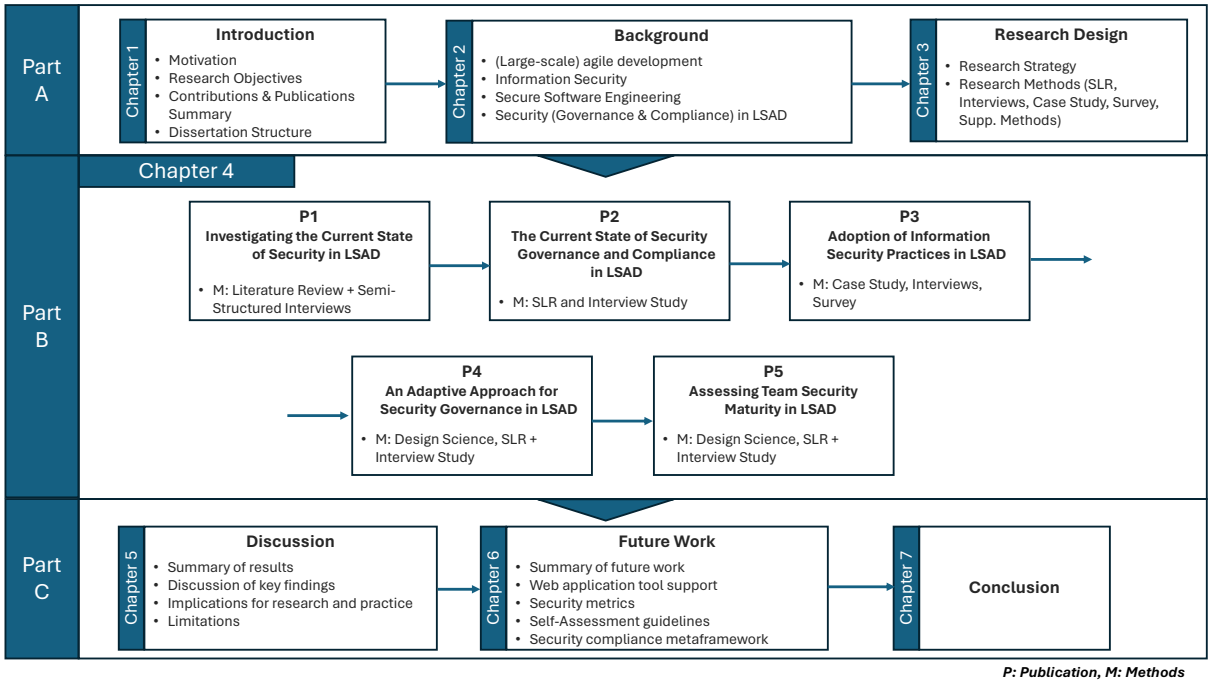

Figure 1.1.: Structure of the dissertation.

CHAPTER 2

Background

In this chapter, we lay the foundation for understanding the key concepts essential to this thesis by outlining the relevant background, thereby setting the stage for our publications and primary contributions. Throughout these descriptions, we define the most important terms, delineate relevant concepts, and provide background and related work to better contextualize our research results.

First, we explore *agile and large-scale agile development* (Section 2.1), followed by an examination of *information security* (Section 2.2). These two areas mark the two principal domains of our research. Subsequently, we describe further relevant background revolving around *security governance and compliance* (Section 2.3), *secure software engineering* (Section 2.4), and *maturity models* in the domain of SSE (Section 2.5).

In the last section of this chapter, we delve into the current dynamics between these areas, providing insights into the integration of security practices within LSAD environments (Section 2.6).

2.1. Agile and Large-Scale Agile Development

We first describe the emergence and development of ASD and then move on to its application in large-scale environments, which is the core study focus of this research.

2.1.1. Agile Development

In the late 1990s, agile methods were developed as a response to the complex, cumbersome processes with long planning cycles, sequential activities, and extensive documentation prevalent in the field of software engineering [HSG18]. Focusing on reducing the heavy-weight footprint of models with longer sequential phases such as the waterfall model [PWB09] or the V-model [FM91], the agile movement emphasizes human collaboration and simpler, more flexible processes [BBC+01, HSG18]. Rather than a single methodology, agile methods encompass a variety of approaches sharing common goals and values, such as self-organization, direct collaboration, iterative development, embracing change, active customer involvement, and lean thinking that aims to reduce waste [CH01, DNBM12].

Conboy [Con09, p. 340] defines agility as a software development method's ability to "[...] rapidly or inherently create change, proactively or reactively embrace change [...]", thus enhancing customer value in terms of cost-efficiency, quality, and simplicity. Independent of a specific agile software engineering method, Kruchten [Kru13, p. 351] broadens this definition to an organization's capacity to respond to "[...] changes in its environment faster than the rate of these changes."

One of the earliest, most famous, and commonly quoted written descriptions of agile methods is the Agile Manifesto [BBC+01], published in 2001. It encapsulates this perspective of modern software development into four core values and twelve principles, advocating for regular, valuable software deliveries and emphasizing teamwork, stakeholder communication, and continuous improvement [BBC+01, DNBM12].

Based on this foundation, several methodologies, such as Scrum [Scr], Extreme Programming (XP) [ext], and Feature-Driven Development (FDD) [PF01], were formalized to apply these values and principles in practice [DNBM12, KNA+16, DFP19].

These methodologies have significantly shaped software engineering practices over the past two decades, with a study revealing that approximately 71% of the 788 surveyed organizations have adopted agile approaches by 2023 [dig24].

Experience has demonstrated that ASD methods offer significant advantages in dynamic environments compared to traditional methodologies [CH01]. The benefits, as highlighted by both academic research and practical application, include improved adaptability to changing requirements and priorities, faster market delivery, reduced project risks and costs, as well as enhanced transparency, predictability, and developer satisfaction, collaboration, and productivity [PW10, dig24].

However, it's important to note that agile methods were initially designed for small, co-located teams working on projects with fluid requirements and are typically most effective in "non-safety-critical projects with volatile requirements, built by relatively small and skilled co-located teams" [WC03, p. 40]. Supporting this perspective, Kruchten [Kru04] identifies the "agile sweet spot" as situations where agile methodologies excel: specifically, when a small, co-located team is developing business applications within the confines of short-term projects.

In this thesis, the terms "agile software development" and "agile development" are used inter-

changeably. However, it is noteworthy that agile methods are increasingly applied beyond software development, such as in general product development. Similarly, "software development" and "software engineering" are treated as synonymous in our thesis, despite the occasional use of "software engineering" to denote more structured development scenarios [AI13]. We believe this approach is justified as the distinction between these terms is not crucial to the core focus of our research, and the differences between these concepts are often very subtle.

2.1.2. Large-Scale Agile Development

While agile methods have traditionally been most effective within the "agile sweet spot," their success and potential benefits have led to their increasingly widespread application across diverse contexts, including safety-critical systems, large organizations, and complex projects [HSG18, Kru13, DPL16, DMFS18]. This expansion raises questions about how ASD can be adapted and scaled beyond these ideal conditions [WC03, Kru04], and it prompts inquiry into how success rates might vary in these broader applications [Kru13].

The term and concept of LSAD is defined and used differently by multiple authors [DM13, DFI14, DPL16]. It may refer to scaling agile practices to larger teams, multiple-team projects, or across an entire organization [DM14]. Various metrics have been proposed to determine what constitutes a large-scale setting, including project costs, duration, the number of people involved, lines of code, development sites, and teams [DM13, DM14].

Ambler [Amb08] extends this definition to consider factors like geographical distribution, entrenched culture, system complexity, legacy systems, regulatory compliance, organizational distribution, and the degree of governance and enterprise focus. Dikert et al. analyzed multiple different definitions and synthesized a definition based on their findings. They suggest that large-scale agile can be defined as "a software organization with 50 or more people or at least six teams" [DPL16, p. 88]. Similarly, Dingsøyr et al. [DFI14] link the definition of "large-scale" to the number of people involved, considering environments with two to nine teams as large-scale. They argue that environments where more than ten teams collaborate represent a very-large scale, necessitating additional coordination mechanisms [DFI14].

2.1.3. Scaling Agile Frameworks

Both research and industry have responded to the increased adoption and demand for scaling agile practices by developing several frameworks that outline structural and procedural guidelines and tools for implementing agile methods on a large scale.

Based on the latest annual "State of Agile Report" [dig24], founded on survey data of 788 respondents in 2023, the "SAFe" [SA24b] is the most widely used scaled approach in industry, followed by "Scrum@Scale/Scrum of Scrums" [Sut]. Other examples of prominent frameworks for scaled agile development are "Disciplined Agile (DA)" [Ins24], "LeSS" [LV16], and the "Spotify Model" [KI12]. Uludag et al. [UKXM17, UPPM21] regard SAFe, LeSS, and DA as the most mature, based on the volume of scientific contributions and detailed case studies. However, these frameworks for scaled agility have been criticized for a general lack of empirical studies,

as well as specific guidelines on their practical implementation and the challenges encountered [KHR18, DCB19]. Research has pointed out that current scaling frameworks often fall short in providing clear directions for security compliance and governance [DCB19], and their effectiveness in meeting compliance and governance objectives is questioned [DPL16, PPL18]. Our analysis [NSM23] supports this, finding that most established scaling agile frameworks do not adequately address these issues, with the notable exceptions of DA and SAFe, which offer some useful guidance but still do not meet all the identified challenges. This thesis aims to address this empirical gap, specifically in the realm of security, with the anticipation that these findings will be partially applicable to other areas of quality requirements as well.

In accordance with other literature [UPP+22], we use the terms scaling agile frameworks, scaled agile frameworks, and LSAD frameworks as synonyms.

2.2. Information Security

Given the broad scope of the information security discipline, which encompasses numerous interrelated, overlapping, and often synonymously used terms and concepts, the following Subsection 2.2.1 aims to delineate and clarify these terms. Moreover, we dive into the most relevant aspect of information security within the scope of this thesis: The domain of software and application security (Subsection 2.2.2). In addition, we describe related concepts that are essential for understanding and addressing the challenges faced by organizations integrating security in LSAD (Subsection 2.2.3).

2.2.1. Information Security, I(C)T Security and Cybersecurity

When discussing security in the area of IT, the terms *information security*, and *cyber security* are often used interchangeably [BBSB17]. For the purposes of this thesis, however, we describe an approach to differentiating between these terms, primarily following the delineation by von Solms and van Niekerk [vv13].

According to von Solms [Sol98], the goal of *Information Security* is to "ensure business continuity and minimize business damage by preventing and minimizing the impact of security incidents" [Sol98, p. 224]. In the realm of Information Security, there are three so-called protection goals that have to be ensured, namely confidentiality, integrity, and availability of information, often referred to as the CIA triangle [BBSB17, SC14, ISOb]. In more simple terms, these protection goals aim to maintain information that is accurate, complete, and accessible whenever needed, while protecting it from unauthorized access [ISOb]. It is noteworthy that there have been continuous efforts to redefine or extent these goals in the past, for example by the additional goal of non-repudiation [SC14]. To achieve these goals and thereby an acceptable level of information security, organizations plan and implement a combination of technical and organizational measures, often also referred to as *security controls* [FGJ14, EP17].

With these definitions in mind, the term information security can be delimited from Information and Communication Technology (ICT) security and cyber security based on the protected assets.

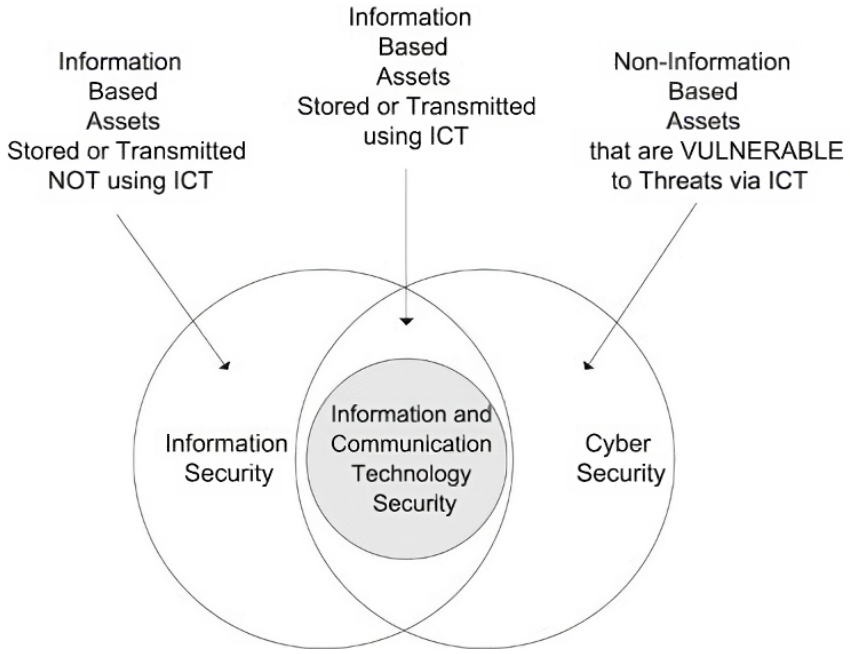

Figure 2.1.: Interrelation between Information Security, ICT Security and Cyber Security by von Solms and van Niekerk [vv13].

ICT security focuses on the protection of information stored and processed within IT systems, including ensuring the proper operations of the technological infrastructure itself, and can be used synonymously with the terms *information systems security* and *IT security* [Ins06, vv13].

In contrast, information security encompasses the safeguarding of all forms of information, not just those processed or stored by technical systems [vv13]. *Cyber security* extends this concept even further by protecting assets accessible via the internet from attacks, including digital information, internet users, societal values, and national infrastructure [vv13]. To distinguish further, cyber security does usually not cover the handling of analog information, i.e., information that is not stored or accessible using information technology [vv13], such as paper documents.

To summarize, ICT/IT security is a subset of information security while cyber security extends the area of information security by protecting more intangible social values. The interrelation of the three terms is demonstrated in Figure 2.1.

2.2.2. Software and Application Security

Since the focus of this thesis is on security in the context of software development, we deem it important to also define the term *software security* and the closely related term *application security*.

The demand for developing secure software products has notably increased, not only in industries with heightened security and privacy concerns like banking and insurance [BBCJ15, BSM20]. This trend is driven by the pervasive and critical role of software in essential aspects of daily

life, including mobile phones, energy supply, and transportation, along with the escalating dependence on software [McG06a, ABE+08].

McGraw [McG06a, p. 14] characterizes software security as "[...] the process of designing, building, and testing software for security", aiming to create applications that are resilient to attacks, quickly recoverable, and continue to maintain their proper functionality during attacks [McG06a, ABE+08]. Application security, on the other hand, can be described as being "[...] about protecting software and the systems that software runs in a post facto way, after development is complete" [McG04, p. 80].

Bell et al. [BBSB17] emphasize the goal of mitigating risks and preventing potential financial, physical, or reputational harm to an organization. To achieve this goal, reactive and proactive approaches should be reasonably combined and integrated into the software development environment [VM01, CJS15]. Proactive security measures aim to identify risks and prevent vulnerabilities preemptively during software development by using methods such as risk assessments, threat modeling, Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) [HN19], security architecture and code reviews, or penetration tests. Reactive measures, on the other hand, aim to identify security issues and attacks during application operations [McG02, McG04, CJS15]. These may include security logging, monitoring, and alerting, e.g. by integrating with security information and event management (SIEM) tools, as well as vulnerability management, incident response, and recovery. However, the distinction between proactive and reactive measures is not always clear, for example, vulnerability management can also be seen as a proactive measure [CJS15] since it aims to rectify known vulnerabilities before they can be exploited in a system by patching them as soon as possible.

Finally, software security can also be seen as a subarea of software assurance that, according to Allen et. al, aims to achieve the "ability to trust that software will remain dependable under all circumstances" [ABE+08, p. 6]. The desired level of software security varies depending on factors like the type of system developed, its target audience, and the system environment, e.g., the industry sector in which the system is being used [BBSB17]. More background on achieving software security through SSE practices is presented in Section 2.4.

2.2.3. Related Concepts

Closely related to the various security terms are the concepts of *risk*, *threat*, and *vulnerability*. According to the ISO Standard ISO/IEC 21827 [ISO08, p. 5], a risk denotes the "potential that a given threat will exploit vulnerabilities of an asset or group of assets to cause loss or damage to the assets". The term threat defines "capabilities, intentions and attack methods of adversaries, or any circumstance or event, whether originating externally or internally" that can generate harm [ISO08, p. 6]. A vulnerability is a flaw or weakness that can be exploited by threats [McG06a, ISO08]. Additionally, the goal of developing secure software is to minimize the number of risk acceptances [BBSB17], i.e. the "informed decision to take a particular risk" [ISO08].

This thesis heavily relies on risk as one of the primary determining factors for balancing the

control and autonomy tension when integrating security in LSAD, advocating for higher team autonomy in lower-risk situations and for more capable teams.

Another concept that overlaps with security is *privacy*. Information security and privacy often go hand in hand, but the fundamental difference is that security focuses on technical and organizational measures that aim to ensure the CIA protection goals of any information deemed worth protecting, whereas privacy revolves primarily around the safeguarding of individual's rights concerning protecting Personally Identifiable Information (PII). Privacy asks questions such as what, why, when, how, and by whom sensitive personal information is collected, processed, stored, and in some cases even transferred to other parties. Privacy also promotes principles such as minimization of data collection as well as privacy-enhancing measures for proper retention of information [Jaj96], such as anonymization through methods like Differential Privacy [DR13]. In our research, we do not explicitly address *privacy* concerns. However, a better integration of security in LSAD also promotes a better foundation for improved privacy. There are also certain security activities that directly benefit privacy concerns, for example, conducting risk assessments during development which typically includes an evaluation of the importance and sensitivity of the data collected, processed, or stored in the application.

2.3. Security Governance and Compliance

Building on the introduction of information security, this section delves into the crucial area of governance and compliance, which is essential for achieving the desired security levels within an organization and, therefore, plays an important role in our research. We begin by exploring the general framework of IT governance and compliance (Section 2.3.1), followed by a focus on security (Section 2.3.2). Subsequently, we discuss relevant security standards and regulations (Section 2.3.3), which are often targeted by governance and compliance initiatives. Finally, we highlight agile and lean governance (Section 2.3.4) due to our research focus on LSAD environments.

2.3.1. IT Governance and Compliance

In order to introduce information security governance, we believe it is beneficial to briefly explain the background of IT governance and compliance, since information security governance and compliance is a sub-discipline of the former.

IT governance is an important part of corporate governance and ensures alignment between a corporation's enterprise and IT strategy, advocating for the responsible use of IT resources and comprehensive IT risk management [Wil01, SS09].

According to Weill [Wei08, p. 3], IT governance can be summarized as "specifying the framework for decision rights and accountabilities to encourage desirable behavior in the use of IT." It consists of "[...] the leadership, organisational structures and business processes, standards and compliance to these standards, which ensure that the organization's IT supports and enables the achievement of its strategies and objectives" [Cal05, p. 20] and the core main areas are IT strategic alignment, IT value delivery, risk management, and performance measurement. This is

also often referred to as "alignment" between business and IT, aiming to ensure business value generation through IT [vDHG06]. For the purpose of alignment, policies, standards, and best practices are created and communicated throughout the organization, aiming to achieve the desirable behavior matching with the strategy goals [LCd+10]. Subsequently, compliance with those policies should be regularly assessed and assured [HSG18].

Initially spurred by growing regulatory pressure and compliance demands, the importance of IT governance has been magnified by increasing IT investments and usage [Dam05, PM12]. Since then, the impact and challenges of IT governance have been well-explored, with researchers reporting that effective IT governance may improve a company's financial performance [WR04]. Other researchers describe advantages such as improved knowledge exchange and decreased IT expenditures through the better business and IT alignment that IT governance aims for [KLHN18].

However, with the increasing prevalence of agile methods, traditional IT Governance efforts are being questioned regarding their effectiveness and suitability for modern IT environments, with adapted approaches such as lean and agile governance gaining a strong foothold, as described in more detail in Section 2.3.4.

This issue is also at the core of one of the main challenges of security in LSAD that we address in our thesis, which is the question of how to balance central control, enacted by an IT governance team or similar function, with the desired autonomy of agile development teams.

An additional vital aspect of IT Governance involves ensuring adherence of IT with legal regulations and self-imposed directives [von05, PM12], commonly referred to as *IT compliance*. Von Solms [von05] highlights IT compliance as an integral part of the broader IT risk management framework within an organization where IT must meet specific requirements, standards, norms, or laws.

2.3.2. Information Security Governance and Compliance

Information Security Governance can be seen as one specific, integral subarea within IT Governance and corporate governance [Pv04, VS01] that revolves around "the integrity of information, continuity of services and protection of information assets" [Wil01, p. 64], in line with the general information security protection goals CIA triad described previously. Therefore, it aims to handle risks that are relevant to the protection of IT environments [von05]. The ISO 27000 standard for "information security management - overview and vocabulary" [ISOa] defines information security governance as the "system by which an organization's information security activities are directed and controlled" [ISOa, p. 4].

In contrast to purely technical aspects of information security, security governance enables organizations to view security not merely as a non-functional requirement, but as a crucial, organization-wide concern overseen by top management through a systematic governance approach, which is deemed essential by numerous researchers [VS06, Pv04, Ins06, CRMO16].

There are also recommendations in literature and practice that suggest dividing Information Security Governance into two distinct dimensions to ensure effective governance processes and

separation of duties. Such a distinction can differentiate between operational management and compliance management, as proposed by von Solms [von05].

The former, operational management, encompasses governance of day-to-day security work, including the administration of security hardware and infrastructure, such as firewalls, access control, or endpoint protection and anti-malware. Nevertheless, it also includes non-technical tasks like establishing information security policies and implementing compliance enforcement mechanisms to assure adherence to all procedures, alongside managing IT risks effectively [von05].

The latter, compliance management within information security, concentrates on enforcing and measuring adherence to set standards, hence identifying and managing IT risks. Tasks under this category include monitoring the progress in addressing identified IT risks, enhancing information security awareness, evaluating the thoroughness of information security policies, procedures, and standards, and ensuring overall compliance with these policies and other regulatory requirements [von05].

Security compliance may also be regarded as a subset of IT compliance, which we have described in Section 2.3.1. In order to now define security compliance more precisely, we turn to Julisch [Jul08] who defines security compliance as the state of adherence to externally or internally imposed functional security requirements, along with the provision of evidence for such conformity. In simpler terms, it can be described as "conformance with a given set of security requirements" [Jul08, p. 71]. These requirements may stem from a mix of international, local, industry-specific, or product-specific standards and regulations [Jul08, MAR+20], which will be described in the subsequent section.

Furthermore, transparency and trust among stakeholders regarding security measures are crucial and can be bolstered through compliance [Tas09]. It underscores a commitment to maintaining security and adhering to relevant security and privacy standards [BBSB17, MAR+20]. The involvement of third parties, e.g., auditors, can enhance this trust [BK04].

Given the increasing reliance on information systems and IT infrastructure within organizations, the significance of information security is clear, underlining the necessity for robust information security governance and compliance [Wil01, von05].

2.3.3. Security Standards and Regulations

As outlined in the previous section, security compliance ensures that organizations adhere to relevant security standards and regulations. Security standards and regulations both aim to safeguard enterprises and their assets by establishing a universal set of requirements or controls. The key distinction is that regulations are legally enforced by governmental or sector-specific authorities, whereas standards are generally voluntary. The terms "controls" (typically used in standards) and "requirements" (commonly used in regulations) serve similar purposes and are used interchangeably throughout this thesis.

An example of a regulatory authority in Germany is the Federal Financial Supervisory Authority (BaFin), which oversees the financial sector, including IT practices of banks and insurance companies. On the other hand, security standards are often created by specialized groups, such

as the National Institute of Standards and Technology (NIST) or the Center for Internet Security (CIS), both based in the United States.

Thus, the sources of security standards or regulatory requirements can be categorized into three main types: international security standards, local security regulations, and industry-specific security standards. Additionally, companies frequently develop their own internal standards and guidelines, typically adapting established industry standards to meet their specific needs.

Organizations typically adopt these standards and regulations to gain structured, comprehensive, and often prioritized guidance on key security controls, thereby fostering trust among entities that develop and use interconnected information systems [MC16]. This is particularly critical in highly regulated sectors, where compliance with legal and regulatory mandates heightens the focus on security, risk management, quality assurance, and traceability [BBSB17].

Compliance with standards or regulations typically involves implementing all controls specified in the relevant documents, with the exception of controls that are explicitly marked as optional inside the standard. Many standards provide for third-party audits where independent auditors evaluate an organization's adherence to these controls through interviews, document analysis and technical assessments. Successful audits result in a certificate of compliance. Compliance may also be accredited without full validation of all controls, but rework on these gaps is then often mandated. Also, compliance does not require the absence of security incidents; instead, it ensures that such incidents are promptly and effectively addressed and thoroughly documented [BBSB17]. Furthermore, while audits represent a snapshot in time, maintaining compliance should be an ongoing effort to ensure standards and regulations are consistently met. Continuous compliance is designed to address security incidents quickly and methodically, focusing on meeting regulatory requirements continuously, rather than adopting a "big-bang" approach where compliance is only ensured immediately before the execution of an audit or a product's final release [BBSB17, FS17].

Concrete examples of security regulations that are mandatory to address in certain industries are the Payment Card Industry Data Security Standard (PCI DSS) [Cou], the Association of the German Automotive Industry Information Security Assessment (VDA ISA) [VDA], or the German Supervisory Requirements for IT in Banking and Insurance Undertakings (BAIT [Autb] and VAIT [Auta]). An additional example is the European General Data Protection Regulation (GDPR), which is primarily focused on privacy and data protection, but also prescribes the use of suitable security measures of technical and organizational nature, as stated in Article 32, "Security of Processing" [PC]. Non-compliance with these regulations risks severe penalties, including the loss of market licenses.

Notable examples for industry standards on the other hand are the ISO/IEC 27001 Information Security Management Systems Standard [ISOb], the NIST Cybersecurity Framework (NIST CSF) [PQS24], the Cloud Security Alliance Cloud Controls Matrix (CSA CCM) [All], or the BSI Cloud Computing Compliance Criteria Catalogue (BSI C5) [Ger]. In addition, there are more governance-focused standards and frameworks that may help to integrate security governance into an overarching IT and corporate governance, such as the international standard for Corporate Governance of IT (ISO 38500 series) [ISOc] and COBIT 5 for Information Security [ISA12].

While compliance with these standards is not legally mandated, they can lead to a competitive advantage since they are increasingly frequently listed by organizations as a prerequisite for conducting business together [Eur16].

Utilizing security standards may benefit even those companies not seeking a specific certification by aiding in security project prioritization or internal control framework development.

However, it is noteworthy that a common pitfall with security regulations occurs when prioritizing compliance over genuine security improvement. Focusing solely on meeting compliance requirements can lead to a false sense of security. Compliance does not equate to absolute security; organizations can be compliant yet vulnerable, or non-compliant yet secure, which is also a reoccurring theme in literature [Jul08, RHL15, BBSB17, MAR+20].

2.3.4. Agile and Lean Governance

A recent study by Horlach et al. [HBSD18], as well as our own research in **P1** [NSM23], show that IT and security governance are still often imposed on development teams by traditional, top-down, authority-led processes through central governance units. Literature similarly depicts information security governance as predominantly authoritative, emphasizing top-down processes and control [Ins06].

This traditional approach frequently results in conflict with agile values and principles [HBSD18], which is one of the key challenges revealed in our research on security in LSAD.

Since information security governance functions are not acting as an end in itself, but aim to ensure information security and compliance [von05, SS09, Fed11], it is not feasible to just forego such a function completely. However, while agile methods aim for increased team autonomy [HBSD18], bottom-up self-governance approaches, and decision-making by technical experts within the team, agile teams are now subject to limitations through governance policies, standards, or best practices [UKXM17]. In addition, the traditional governance models are seen as impeding and slowing down agile teams by imposing bureaucratic restrictions [UKXM17].

To mitigate these issues, both academic literature [Amb09, PKM+21] and multiple scaling agile frameworks such as SAFe [SA24b] and DA [Ins24] propose transitioning to modern governance approaches such as lean and agile governance.

Agile governance is a broad term and even extends beyond software engineering to various disciplines such a manufacturing, integrating the principles of agile philosophy into traditional IT governance frameworks [LCd+10, JOKE+14, LKRM16]. Given our research focus on governance within (large-scale) ASD settings, Gill's [Gil07] definition is particularly relevant. According to Gill [Gil07], integrated agile governance is characterized by being lightweight, collaborative, and communication-oriented. It is also cost-effective and capable of adapting over time, featuring efficient frameworks, controls, processes, and structures. It aims at enhancing business value through the strategic alignment of business and agile objectives, alongside effective performance and risk management [Gil07].

Lean governance is predominantly discussed within industry publications, including white papers and scaling agile frameworks [AKS11, AL19, SA24b]. It merges traditional IT Governance

with a value-driven, waste-reducing style of working [AKS11, PKM+21], fostering decentralized and consensus-oriented decision-making [UKXM17, KLHN18]. It aims to harmonize agile principles with the requisite compliance processes [PKM+21]. Central to lean governance is the empowerment and encouragement of development teams. This is achieved through collaborative and supportive methods, incorporating practices like risk-based milestones, embedded compliance by integrating governance in each phase of development during iterations, and adopting a pragmatic stance by governance bodies [Amb08, Amb09, TD09, AKS11].

Even though agile and lean approaches differ in their manifestations, their fundamental principles align closely and can be effectively combined [JOKE+14, MBK+18]. Both challenge the traditional IT governance model characterized by authority, coercion, and top-down decision-making [LB18]. Instead, agile and lean governance emphasize autonomy, self-organization, and bottom-up engagement, fostering a more collaborative governance style [AKS11, LKRM16, AL19].

Although empirical research on the practical implementation of agile and lean governance is limited, its significance is widely acknowledged [VRC19]. A consensus among scholars suggests that integrating information security within lean and agile frameworks is both feasible and essential, and that empowering autonomous, mature development teams can offer several advantages for companies [KNA+16, MAR+20, NAD20]. For instance, Vejseli et al. have observed that, similar to traditional IT governance, agile and lean IT governance positively impacts business-IT alignment, thereby enhancing enterprise performance [VRC20]. Another key advantage is that teams that manage and operate their services independently tend to achieve significantly higher levels of speed and efficiency [BBSB17]. Ambler and colleagues advocate for an agile and lean approach to IT governance and compliance to foster scalability in software development [Amb08, BAR13]. Furthermore, agile and lean governance may promote organizational commitment across all divisions, facilitating increased business agility [JOKE+14, LMd20].

In addition, as the quality of their products improves, these teams may require fewer and less labor-intensive compliance checks, easing the burden on central governance units [PKHR21]. When it comes to security compliance and auditing, Wright [Wri14] offers one of the most comprehensive contributions to the domain of agile governance, highlighting the importance for organizations to acknowledge the distinctions between traditional and agile projects, as well as the cultural differences between auditors and agile teams, to establish effective governance in agile settings.

Drawing from all the previously described background and related work, which demonstrates the clear benefits of agile and lean security governance and the need therefor, we aim to transfer the relevant findings to our research context of security in LSAD. Furthermore, we aim to leverage existing bottom-up aspects of security governance. The goal is to leverage bottom-up approaches and empower agile teams, but through stringent and systematic ways that include all necessary security and compliance considerations and are audit-proof, which we are aiming for with the solution artifacts of our research. Our solution approaches in **P4** and **P5** represent a form of agile and lean governance, thereby helping to integrate and establish security in large-scale environments.

According to scaling agile frameworks such as SAFe, lean governance should avoid traditional quality gates and extensive documentation requirements. Instead, it emphasizes the integration

of quality early in the product development process. This approach is encapsulated in one of SAFe's core principles, stating to aim for value flow without interruptions [KL20], advocating for continuous value delivery without inspections that stop the development flow [KL20, SA24a].

2.4. Secure Software Engineering

In the following, we outline the existing background on security in software engineering, so that we can transfer valuable aspects to security in LSAD contexts, since many of the established process models, frameworks, and standards for secure software development are not or only partly designed for agile or lean environments [BC11, BS15], and especially not for LSAD environments and their resulting challenges.

Secure Software Engineering (SSE) encompasses the comprehensive integration of practices, activities, and processes aimed at ensuring the security of applications throughout the Software Development Lifecycle (SDLC). SSE, sometimes also referred to as *Security Engineering* or *Secure Software Development*, involves a broad spectrum of tasks including security requirements engineering, risk assessments, software security testing, security architecture, and the establishment of security objectives and policies as well as incorporating secure design principles and guidelines [ABE+08, Han10, RHL18, KKKI21, RRH+21].

Hence, SSE is directly connected to the concepts of software and application security presented in Section 2.2.2, with SSE having a particular emphasis on the process involved in building secure software and thereby achieving software and application security.

A prevalent perspective in both academic research and industry practice is that security should be integrated from the earliest stages of the agile SDLC, and then continuously extending through requirements definition, design, coding, testing, deployment, and maintenance phases [KZ09, Bar11, BBSB17, RRH+21], described with terms and principles such as "security-by-design" and "shift-left security" [SJD19, WMW22].

According to Hadavi et al. [HSSH08], SSE addresses this perspective by involving "well-structured processes" [HSSH08, p. 866] that incorporate security considerations from requirements analysis to design and implementation. By proactively addressing security from the beginning, SSE embeds security as a core element of software development.

This strategy counters the method of retrofitting security after the software product is almost complete, which often results in increased costs and unresolved security vulnerabilities [KZ08].

Various models and international standards have been developed to achieve this aforementioned goal of integrating security into the software development processes.

The Microsoft Secure Development Lifecycle (SDL) is one such widely recognized model. The SDL aims to minimize security vulnerabilities in source code, software design, and enhance security documentation by implementing process improvements and best practices [Mic]. It is structured around eight core principles and five stages, bookended by mandatory security training before implementation and a security incident detection and response strategy following software release.

Additionally, Microsoft introduced SDL Agile, a variant designed to combine traditional SDL practices with agile methodologies, ensuring compatibility with the principles of both. This approach particularly focuses on agile-compatible activities such as threat modeling and the iterative management of security requirements [Sul19]. It also introduces the idea of so-called "buckets" that are used to prevent a large conglomerate of mandatory security activities before every release, which would not work well with the shorter sprints commonly applied in agile development. These are explained in more detail in Section 2.6.4, together with other promising solution approaches that we build on for our own solution artifacts.

Another example are the software security best practices by McGraw [McG04, McG06b], which also emphasizes the proper integration of security best practices into the SDLC. McGraw advocates for embedding security measures like drafting security requirements and abuse cases directly into the development process. This model is adaptable to iterative development methods, allowing for security activities to be repeated each cycle. Moreover, McGraw underlines the critical importance of raising security awareness, education, and targeted training for developers [McG03, McG04].

All these SSE methods, practices, and actions that aim to improve the security posture of a software artifact or the security capability of one or more development teams are what we repeatedly refer to in the thesis as *security activities*. Examples include security requirements management, risk analysis and threat modeling, penetration testing, participating in bug bounty programs, or secure code reviews. We provide a comprehensive overview of relevant security activities in **P3** and, with a slightly smaller scope, also in **P1**.

2.5. Maturity Models

In addition to the security regulations and standards outlined in Section 2.3.3, maturity models for developing secure software play a significant role. While these could be considered under security standards, their relevance to our research, particularly with our publication **P5** introducing a maturity model as one of the key contributions, warrants a separate description of their background.

Nevertheless, we regard them as a distinct subset of security standards, primarily used by organizations not for external certification purposes but to internally evaluate and enhance the maturity of their software development processes. Such evaluations help identify areas for potential improvement and enable benchmarking against the maturity levels of other organizations.

Prominent maturity models in software engineering include the System Security Engineering Capability Maturity Model (SSE-CMM) [Ham99], the Open Web Application Security Project (OWASP) Software Assurance Maturity Model (SAMM) [OWAb], the OWASP DevSecOps Maturity Model (DSOMM) [OWAa], and the Synopsys' Building Security In Maturity Model (BSIMM) [Syn]. A common feature of these models is the categorization of requirements into various domains or dimensions and the classification into different levels of maturity. We will briefly describe each of the mentioned models in the following.

The SSE-CMM [Ham99] sets the international standard for secure software development, out-

lining essential qualities for effective security engineering [ISO08]. Its primary aim is to assist organizations in enhancing their ability to create secure and reliable products, improving quality and availability while also taking costs into account. The model determines 22 process areas, categorized into base practices as well as project and organizational practices, comprising a total of 129 security practices that span the entire security engineering lifecycle and organizational interactions, internally as well as externally. It also establishes five capability maturity levels to evaluate and identify areas for enhancement within the security engineering process.

The OWASP SAMM [OWAb] offers a method and model for analyzing and enhancing the security posture of an organization's software initiative. SAMM is designed to be adaptable and risk-oriented, suitable for various organizational types. It covers the entire software lifecycle and aims to be neutral regarding the used technologies and applied methodology. Version 2.0 of OWASP SAMM introduces five business functions: Governance, Design, Implementation, Verification, and Operations, each with three security practices and three maturity levels respectively, totaling 45 objectives.

The DSOMM [OWAa] serves as a guide for enhancing DevOps environments, focusing on more detailed implementation aspects compared to the higher-level concerns of OWASP SAMM which also addresses governance and compliance. The latest DSOMM version categorizes 156 controls into four maturity levels across five dimensions: Build and Deployment, Culture and Organization, Implementation, Information Gathering, and Test and Verification. It proposes a structured and prioritized approach to applying these controls by distributing them to multiple maturity levels instead of introducing maturity levels per control or practice.

An additional model that we would like to mention is the Security Belts [App24] model, which is partly inspired by DSOMM and SAMM. Although relatively new and not yet widely recognized, it is closely related to our research since it structures security capabilities and evaluates a team's maturity in secure software development. It adopts a nine-level belt system to represent different stages of maturity, with each belt encompassing specific security activities or tasks required for advancement, similar to a checklist. Instead, our approach examines maturity levels within different activities. In addition, while the Security Belts model provides structured assessment guidance, it lacks flexibility in adapting to varying product risks and organizational contexts. Our research, particularly in publications **P4** and **P5** extends the ideas revolving around security maturity by situating them within a broader context. Our approaches aim to enhance practicality and applicability by considering product risk and organizational factors, offering a more versatile approach to security maturity assessment.

Finally, the BSIMM [Syn] is an annual industry survey analyzing software security practices across roughly 130 organizations. Originating from a 2008 study by the now-called Synopsys Software Integrity Group, BSIMM has evolved into a descriptive model that benchmarks typical activities in software security initiatives. Its latest version, BSIMM14 [syn23], identifies three maturity levels and structures 126 activities within four domains - Governance, Intelligence, SSDL Touchpoints, and Deployment - each divided into three practices covering areas like code review, risk analysis, penetration testing, attack modeling, and training.

The standards just described above do not specifically cater to agile and large-scale agile envi-

ronments. However, certain observations, findings, and recommendations from these standards can be adapted for security integration within scaled agile product development contexts.

One of the main gaps identified and addressed in our research concerns the existing models' focus on organizational capabilities rather than providing a direct means to assess the security maturity of individual agile development teams. This specific gap is what we aim to fill with the approach proposed in our publication **P5**. In addition, our security activity approach proposed in **P3** can be applied to evaluate suitable practices from these models, especially for use in LSAD contexts. Furthermore, our models in **P4** specifically address the challenges of security in LSAD, going beyond the existing models.

2.6. Security (Governance and Compliance) in (Large-Scale) Agile Development

In the following sections, we will integrate the topics previously presented, focusing on the interplay between security, governance, compliance, and LSAD. Specifically, we will explore the numerous challenges encountered within this interplay in Section 2.6.1. Following that, we will examine current solution and integration approaches designed to mitigate some of these challenges. There are multiple possible ways to categorizing the existing approaches. In **P3**, based on the collected data, we distinguish between role and knowledge approaches versus methodology approaches. However, for Sections 2.6.2 to 2.6.4 of this thesis, we organize the theoretical background in three distinct categories that we identified and applied in our research presented in our publication **P2**.

2.6.1. Challenges

Academic literature widely recognizes the inherent areas of conflict between information security, IT governance, and the principles of (large-scale) agile software engineering [Bar11, FSOO13, ORbO15]. This discord has already been identified and highlighted by researchers over a decade ago. One example is the exploratory study by Bartsch, who performed semi-structured interviews on security challenges in agile development contexts and related mitigation strategies [Bar11]. In more recent studies, Riisom and colleagues [RHA+18] along with Bishop and Rowland [BR19], acknowledged the longstanding issue yet noted the ongoing lack of satisfactory solutions. The conflict is further underscored when examining the Agile Manifesto, a foundational text of agile development, which advocates for prioritizing "value individuals and interactions over processes", "collaboration over contract negotiation", and "responding to change over following a plan" [BBC+01]. These values contrast sharply with the stringent requirements of security standards and regulations, which mandate comprehensive documentation of responsibilities, agreements, and established processes, thus highlighting the significant tension between these domains.

Despite the recognized problem, organizations often "end up running a non-agile security development life cycle along the agile software development processes" [RHL18, p. 1]. This approach is counterproductive to the goal of ASD, which focuses on delivering functional products swiftly

and frequently through short cycles and constant adaptations to frequent changes [BBC+01]. On the other side, such a focus on functionality tends to sideline non-functional aspects like security and other quality attributes [NSBJ10, BS15].

Additionally, the emphasis of agile methods on early and frequent software delivery is challenged by the need for extensive documentation and traceability to achieve software that is compliant to industry, regulatory, or self-imposed standards, which is typically assured and validated by several stakeholders such as internal governance and compliance functions or external auditors [BK04, FSOO13, Bas16].

Furthermore, the more security requirements organizations face, the more likely they prefer stability over change, as security activities, traditionally designed for less iterative processes, may yield suboptimal outcomes when applied in an agile context [AN16, SS20]. Hence, in some scenarios, security may hinder reaping the benefits of agile development, and agility may deteriorate security aspects, potentially leading to overlooked security breaches or inadequate responses [BBSB17, BJBC17, Bar11, BBCJ15, vBS18]. The tension between agility and security becomes even more pronounced when scaling agile practices, particularly when trying to maintain security compliance without compromising speed and flexibility [MMBK21]. In large-scale agile settings, achieving security compliance is notably challenging due to the complexities of governing multiple teams and coordinating security measures. Van der Heijden et al. [vBS18] identify three specific security challenges in LSAD: Aligning security objectives across distributed teams, clarifying roles and responsibilities in security activities, and integrating security testing tools with minimal overhead.

Addressing these challenges and aligning security objectives and establishing common guidelines and quality standards to ensure technical consistency and effective collaboration across teams usually requires some form of governance that may conflict with the goal of autonomous, self-organizing teams in agile environments [BAR13, Bas16, BH19].

A recent study by Horlach et al. [HBSD18] reveals that organizations still frequently employ a traditional, top-down, authority-led approach, imposed by central governance units on development teams. This method conflicts with agile's principle of empowering autonomous development teams, free from cumbersome bureaucracy, favoring a bottom-up, self-governance style focused on more autonomy [HBSD18, RRS+20]. We have confirmed these findings in our own research. For example through **P1** - involving interviews in nine different organizations - we found that top-down governance is still the norm in all organizations we studied, underscoring the tension between control and autonomy when integrating security into LSAD.

This leads to one of the core challenges we tackle with our research: the control autonomy tension in LSAD security efforts. When using the term *team autonomy*, we describe "self-organizing teams" or "self-managing teams", as defined by Stray et al. in their research on autonomous agile teams [SMH18].

Despite the emergence of practices like *DevSecOps*, a strategy that extends DevOps principles and values by security aspects, which contributes to tackling scaling issues of agile security by heavily relying on automation [MZD+20], challenges persist. For instance, the separation of duties between developers and operators that is mandated by various standards and regulations

(for example when releasing new software iterations to production), clashes with DevSecOps practices that aim to combine efforts in one responsible team.

We provide a detailed analysis and presentation of reoccurring challenges of security in LSAD in our publications **P1**, **P2** and **P3**.

P2 offers an extensive, in-depth analysis of current challenges, which resulted in identifying 15 relevant challenges across these three categories: First, security challenges that are unique to LSAD environments. Second, selected security challenges in (small-scale) agile development which are still relevant in LSAD, and third, selected general challenges in LSAD, transferable to a security context.

One concrete example of a crucial challenge is that organizations face an unclear and heterogeneous understanding of roles and responsibilities regarding security in LSAD environments [Bar11], which we address with our contributions, as explained in more detail in Section 1.3.2.

In our research, we also build on previous studies regarding security challenges in LSAD of other researchers, which we briefly summarize in the following.

Moyon et al. [MAR+20] conducted a systematic mapping study and reported that ensuring security compliance is often tedious for organizations using agile methods as it is often associated with high efforts, which intensifies by decreasing cycle times.

While most literature focuses on the general tension between agility and security or general LSAD challenges, van der Heijden et al. particularly investigated security challenges in LSAD, as mentioned previously. They identified that security challenges in LSAD either stem from (i) general security challenges, (ii) security challenges in small-scale agile environments, or (iii) are unique to large-scale agile such as, e.g., the "alignment of security objectives in a distributed setting" [vBS18, p. 4].

Edison et al. also identified several security-related challenges when introducing scaling agile frameworks in practice, e.g., lacking security awareness and missing proactive conduction of security activities [EWC21], which supports van der Heijden et al.'s [vBS18] assumption that general challenges can also be transferred to the LSAD environment.

2.6.2. Enhanced Large-Scale Agile Approaches

As explored in the previous sections, scaling agile frameworks lack integration of security activities. Also, our framework analysis in **P2** revealed that they provide no or insufficient concrete guidance on governance and compliance, especially when it comes to specific challenges such as the autonomy-control tension, mentioned and described previously by multiple other researchers [HBSD18, NR21, MŠPL21].

The approaches we present and discuss next also identified this gap and aim to reconcile LSAD with aspects related to security, governance, or compliance. Therefore, they represent important related work for our thesis.

One influential approach to our research is the teamwork quality model applied in LSAD environments by Poth et al. [PKR21, PKMR23]. Poth et al. present a maturity model to evaluate

team performance [PKR21, PKMR23] and a related systematic approach [PJR20], encapsulated by the principle that "team maturity leads to higher team autonomy" [PJR20, p. 197]. Based on that principle, they aim to ensure compliance while "offering as much autonomy to agile teams as possible" [PJR20, p. 192] and not impeding software product delivery. The model allows for the tailoring of quality governance, including standards and guidelines, according to the maturity and capability of an agile development team - a concept quite similar to the approach we explore in publications **P4** and **P5**. They recommend conducting team self-assessments every six to nine months, enabling teams to independently utilize the model within a secure, internal setting. For additional guidance, the option to consult team-external coaches is available. The effective use and outcomes of the model are verified during agile retrospectives, with input from external team members connected to the project. This process ensures that teams can reflect on and enact improvements based on learned experiences, reinforcing the model's role as a tool for continuous team enhancement.

They highlight several benefits of their model: it enhances transparency regarding team maturity, facilitating the identification and execution of actions for improvement. This, in turn, bolsters a team's capabilities and efficiency, enhances reliability, minimizes the likelihood of low-quality outcomes, and prevents the unnecessary accumulation of skills unsuitable for current needs.

Although Poth and colleagues primarily address broader software quality aspects rather than focusing specifically on security, their work offers elements applicable to our research. For example, we have adapted the foundational structure of their model, which organizes concepts into pillars, domains, and topics, into our TSMM detailed in publication **P5**. Unlike their use of pillars to further categorize domains, we employ these pillars to differentiate between multiple assessment approaches instead of relying solely on self-assessments. Furthermore, their model resembles a checklist, expanding the activities under each topic as it advances. In contrast, in our approach, a "topic" represents a distinct capability or process, the maturity of which is to be evaluated, rather than just a set of activities to be completed. This difference highlights our model's focus on assessing and enhancing specific areas of security capability or maturity within teams.

Another important related work for our research is an approach developed in the context of a SAFe implementation in the financial industry, the "Earn Your Wings" introduced by Petit and Marnewick [PM19]. This method evaluates the autonomy of agile teams based on their specific context, consequently assigning levels of accountability corresponding to their assessed autonomy. Inspired by the tiered licensing of pilots, which involves five levels of qualification, their work emphasizes the creation of a lean, adaptable governance framework that maintains compliance. The method seeks to streamline the conventional deployment governance, characterized by multiple approval layers before software releases, to adapt to scaled agile environments more effectively. According to Petit and Marnewick, this streamlined process not only accelerates the release procedure but also reduces the efforts agile teams expend on navigating bureaucratic hurdles due to their enhanced accountability [PM19].

The concept of fostering lean and adaptive governance while ensuring adherence to regulations aligns with our research interests. We explore the application of a similar adaptivity within our approach published in **P4**, focusing on the specific parameters that could govern security practices in our research context.

Finally, the work of Moyon et al. [MBK+18] stands out by being one of the few existing publications that explicitly aims to improve the security in LSAD integration, making it highly relevant to our research. They demonstrate how security standard requirements can be anchored in an agile development process and apply this method to a SAFe context where the goal is to achieve compliance with the IEC 62443-4-1 standard describing the secure development of products used in industry automation and control systems. They follow up their work with a revised model named S2C-SAFe [MAR+20] that further enhances SAFe by incorporating additional roles, activities, and artifacts focusing on security requirements and validations.

In relation to our research, especially the idea of enhancing a scaling agile framework by security considerations is interesting, which we also aim for with our results, just in a more framework-agnostic way, as described in **P4**.

2.6.3. Enhanced Small-Scale Agile Approaches

In this section, we explore small-scale agile methods adapted to meet security needs, which, while not explicitly addressing scaling issues, offer insights into how some of these solutions might be adapted and transferred to the large-scale context of our research.

Bell et al. contribute comprehensively to the insights in the research area with their book titled "Agile application security". They argue that developing secure software using agile methods is feasible if agile teams undertake security activities with proper support and guidance from security experts. They suggest that having a team member dedicated to security and ensuring product owners prioritize security and compliance can effectively integrate security into development processes. Furthermore, the adoption of security tools, such as automated testing, allows developers to independently assess their software's security, transforming security into a facilitative rather than obstructive element [BBSB17].

The research results of Newton et al. [NAD20] demonstrate twelve critical success factors and practices for incorporating information security into ASD. They highlight the importance of security awareness among employees, advocating for basic security training for all employees, and defining and implementing explicit security processes. They also recommend early security testing and simple yet secure release processes.

Bartsch [Bar11] underscores the significance of creating a mutual understanding of roles and integrating assurance and documentation seamlessly into agile workflows to merge security with agile methods. He also highlights that agile development has certain strengths that security integration can build on, such as the focus on people and their expertise and the drive for continuous improvement, resulting in his recommendation to focus on security training and spreading awareness.

Boldt et al. [BJBC17] delve into the synergy between security and ASD in their work "Introducing a Novel Security-Enhanced Agile Software Development Process", publishing their research results at Ericsson. They emphasize the necessity for organizations to incorporate security measures systematically throughout the development phase to manage risks efficiently.

Their approach involves embedding security activities, such as risk analysis, into the development

workflow. It also involves equipping developers with security analysis tools and aiming for high-quality results in a cost-effective manner. By integrating new security roles, activities, and protocols into Ericsson's agile development process, they formulated the Security-Enhanced Agile Software Development Process (SEAP) [BJBC17]. The implementation of SEAP at Ericsson demonstrated a positive influence on risk management while keeping security costs manageable [BJBC17].

The security-specific roles and activities identified by Boldt et al. [BJBC17] are particularly relevant to our research. Their successful application in a small-scale agile setting at Ericsson prompts us to explore their adaptability to LSAD environments. Through our publications **P3** and **P4**, we examine the potential of these security measures to fit within a broader, more generic approach.

One additional model that crucially contributed to the creation of our solution artifacts is the three Lines of Defense (LoD) model. Originally developed by the Federation of European Risk Management Associations (FERMA) [Fed11] and the European Confederation of Institutes of Internal Auditing (ECIIA), this model outlines the collaborative roles of various functions such as risk management and internal control. FERMA/ECIIA [Fed11] suggests its application for managing cyber risks, aligning with our research focus. Wright's [Wri14] exploration of the model's application within agile environments has been particularly influential in our research, guiding the establishment of our role and collaboration model in our publication **P4**.

Finally, two models intersect with our research that focus primarily on safety-critical environments. Since security is crucial in safety-critical software development [BMSS18], the findings are also relevant to our research focus.

The R-Scrum model by Fitzgerald et al. [FSOO13] adapts traditional Scrum to meet regulatory requirements by integrating specific enhancements, such as templates for developers and peer code reviews, and introducing a dedicated quality assurance role to ensure ongoing compliance [FSOO13]. Their findings suggest that with proper adaptations and tools, agile methods can indeed function within regulated environments, allowing for regular alignments and validations.

Similarly, the SafeScrum model by Hanssen et al. [HHS+16] modifies traditional Scrum for developing certified safety-critical software, accommodating the IEC65108 standard. This includes implementing dual backlogs for functional and safety requirements and integrating additional XP techniques, change impact analysis, continuous integration, and automated testing tools. A quality assurance role is also introduced to facilitate collaboration and ensure the quality of safety requirements [HHS+16].

Part of our research explores how these safety-focused methodologies could be adapted for security in large-scale environments. Steghöfer et al. point out that both SafeScrum and R-Scrum do not cater to "systems with mixed criticality" [SKHW19, p. 351], leading to potential unnecessary costs for non-critical parts. Our research addresses this by incorporating various influencing factors to enhance adaptivity in our approach, as discussed in **P2** and **P4**, aiming for a model that is both effective in security-critical contexts and adaptable to large-scale agile environments.

2.6.4. Secure Software Engineering Practices Adapted to Agile Environments

Finally, we will now expand on the SSE practices introduced in Section 2.4 and SSE maturity models outlined in Section 2.5, focusing on integration strategies specifically designed for agile development methodologies.

In general, Rindell et al. [RHL18] acknowledge that aligning SSE activities with agile methodologies presents significant challenges. Despite the high cost associated with integration efforts, they emphasize that incorporating security processes directly into agile workflows is more advantageous than operating non-agile security processes alongside agile development activities [RHL18]. This integration should be implemented across three key areas: training individuals, managing security requirements, and incorporating security activities, tools, and experts into the ASD workflow. With sufficient initial planning, security-related tasks should be added to the product backlog and addressed at suitable times throughout the iterative development process [RHL18].

OWASP has developed SAMM Agile [OWAb], a set of guidelines to aid the integration of the SAMM with agile practices, despite its approach-agnostic design. These guidelines detail methods for embedding security measures within agile sprints and facilitating effective teamwork between security experts and developers. For that purpose, they, for example, suggest assigning a Security Champion (SC) to foster autonomy within teams. The model's authors also underscore the significance of reducing manual security testing effort through test automation and advocate for employing security metrics to enhance the maturity and result quality for agile teams. Furthermore, SAMM Agile highlights the adaptation of threat modeling to suit agile cycles, in line with other models such as the Microsoft SDL Agile, presented in the next paragraph.

To address the resource-intensive demands of the Microsoft SDL, Microsoft has developed an agile variant of their SDL. This variant recognizes that fulfilling every SDL requirement in each release or sprint is impractical due to the substantial resource and time investment required. Therefore, SDL requirements are categorized into three groups: onboarding requirements to be implemented within a grace period from the project's inception, every-sprint requirements to be addressed in each sprint, and other requirements categorized into buckets such as security verification, design review, and response plans [Sul19]. For example, a complete threat model does not have to be created for each iteration but only for newly developed functions.

Agile teams are mandated to address each requirement at least once annually, avoiding repetition of the same requirement in consecutive sprints. Beyond these rules, teams have the flexibility to select the requirements they wish to meet [Sul19].

In our research, a critical point of interest is identifying which activities are deemed highly resource-intensive as well as how difficult they are to implement and how much they are aligned with the goals of LSAD environments, as analyzed in **P3**. Implementing varying frequencies and responsibilities for executing specific activities might offer a pathway toward more adaptive and streamlined governance processes. We warranted this aspect with further exploration, resulting in the adaptive process published in **P4**, partly building on the ideas of security activity buckets.

CHAPTER 3

Research Design

In this chapter, we describe the research design we developed to conduct our research and answer the RQs presented in Section 1.2 of this thesis.

In Section 3.1, we present the overall research strategy that we are using. Subsequently, in Section 3.2, we detail the research methods that we applied as part of our overarching research strategy.

3.1. Research Strategy

To achieve the research objectives defined in Section 1.2, our research strategy primarily relies on the DS research method [HMPR04, PTRC07]. DS enables a systematic exploration of the problem domain, facilitating the construction of a robust knowledge base, while considering practical challenges and solutions, leading to the creation of novel, innovative artifacts [HMPR04, PTRC07].

Hevner et al. [HMPR04], who present adaptive guidelines for DS in information systems, describe these artifacts as solutions that tackle either previously unsolved issues or known problems more effectively or efficiently [HMPR04]. These artifacts are intended to be specifically designed to address significant, relevant business challenges so that they are both fit for purpose and useful [HMPR04].

This approach is particularly relevant to our work, as it aligns with our identification of significant research gaps and specific, yet unaddressed, challenges in our field, despite the presence of some initial solution concepts.

Peffers et al. [PTRC07] contribute significantly to the field of DS research by analyzing seven

DS models and deriving a generalized DS process for the field of information systems. They propose a structured approach, aiming to improve the applicability of DS and serving as a guide for researchers.

The consolidated process encompasses six phases [PTRC07]:

1. Problem Identification, where challenges are initially identified;
2. Objective, defining the objectives for solution approaches;
3. Design and Development, where the artifacts to address the challenges are created;
4. Demonstration, demonstrating the functionality and application of the artifacts;
5. Evaluation, assessing the artifacts in a rigorous manner;
6. Communication, disseminating the findings and contributions to both academic and practitioner communities.

This approach ensures a thorough and iterative exploration of the problem space and enables the development of solution artifacts, allowing for continuous refinement and validation of the results.

In our research, we utilize the guidelines established by Hevner et al. [HMPR04] and Peffers et al. [PTRC07] as a foundational framework. However, we tailor the methodology to fit our specific needs and integrate supplementary research techniques.

Notably, we diverge from Peffers et al.'s distinct separation of the demonstration and evaluation phases [PTRC07]. Instead, we merge these steps by showcasing our solution through expert interviews while simultaneously gathering immediate feedback, achieving both a demonstration and assessment of our solutions in real-world contexts. A qualitative expert interview study goes beyond the evaluation methods initially proposed by Hevner et al. [PTRC07]. However, it is in line with the expert evaluation method outlined by Peffers et al. [PRTV12], where artifacts are evaluated by "one or more experts" [PRTV12, p. 402].

This evaluation methodology is particularly fitting for our research project as it complements the use of interviews conducted in earlier phases to discern business requirements and identify current challenges. Through this adapted approach, we facilitated iterative refinements of our solution artifacts, ensuring a thorough and responsive development process that aligns with both academic rigor and practical relevance.

Our research strategy resulted in the following concrete artifacts:

1. A categorization of security in LSAD integration best practices in four dimensions (**P1**)
2. A generic organizational set-up describing the collaboration structure of agile teams in LSAD with security-related roles (**P1** and **P4**)
3. Influencing factors to balance the control autonomy tension (**P2**)
4. An overview of 15 security in LSAD challenges in three categories (**P2**)
5. An evaluation of 14 agile security approaches with their advantages and drawbacks (**P3**)

6. Generic evaluation criteria and typical drivers and obstacles when adopting security activities in LSAD environments (**P3**)
7. An adaptive collaboration model for security compliance in LSAD providing detailed information about security-related roles and respective activities (**P4**)
8. A scoring model to determine the adaptivity of the collaboration model (**P4**)
9. A projection of the model onto the most relevant large-scale agile frameworks to simplify adaption for organizations (**P4**)
10. Team security maturity criteria to evaluate the capability of agile teams to develop secure and security-compliant software (**P5**)
11. An exemplary team security maturity model based on the identified criteria (**P5**)

It also resulted in multiple avenues for ongoing and future research and solution artifacts, as summarized in the research outlook in Chapter 6.

There are two important concepts in our DS research endeavor leading to these solution artifacts: *rigor* and *relevance*.

Rigor involves the methodical construction of a knowledge base through scholarly methods, while relevance pertains to the practical applicability and impact of our findings in real-world settings [HMPR04]. To uphold rigor and systematically develop a relevant knowledge base, we conduct multiple SLRs, adhering to the guidelines and best practices outlined by Webster and Watson [WW02], Brocke et al. [BSN+09], as well as Kitchenham and Charters [KC07].

For ensuring relevance, we utilize interview studies, primarily following methods and guidelines suggested by Myers and Newman [MN07], Rubin and Rubin [RR12], and Saldana [Sal15], to uncover business needs and assess practical applications.

The nature of the research presented in this dissertation is primarily qualitative-empirical and constructive, focusing on designing solution artifacts grounded in collected data and a thorough analysis of the problem domain — the integration of security in LSAD. However, for our initial analysis of security in LSAD, which supports our first research contribution and addresses RQ1, we primarily employ inductive research to infer patterns from observations and theoretical constructs [Bha12].

Our methodology predominantly employs qualitative techniques such as expert interviews and case studies, since they are suitable to deeply explore and evaluate a complex phenomenon [Bha12, SC90]. To augment our qualitative focus, we also incorporate quantitative methods, which typically aim at measuring variables to identify patterns and test hypotheses [Yil13, CC17]. In particular, we conducted a survey as part of our case study in **P3** to assess agile security practices. In addition, we used quantitative aspects in our qualitative collection methods. For example, also in **P3**, we asked interviewees to rate security approaches within a numerical scale, in addition to their qualitative answers. This combination of qualitative and quantitative methods positions our research within a mixed-method framework, though with a stronger emphasis on the qualitative side.

The primary data collection techniques that we employ and integrate into the overarching DS

research process include systematic, multivocal literature analysis, document analysis, and semi-structured expert interviews. This multi-source strategy, also referred to as triangulation, gathers insights from diverse sources and through various methods, enhancing the validity of our findings [Sea99, BH06]. For data analysis, we employ mixed methods, both inductive and deductive, primarily drawing from established qualitative content analysis techniques as recommended by Kuckartz [Kuc14] and Mayring [May00].

Our application of these methods in our research is described in more detail for each method in the following Section 3.2.

To ensure the proper conduction of these methods, we are guided by the recommendations for empirical research proposed by the ACM Special Interest Group on Software Engineering (SIGSOFT) [SIG24].

To summarize, our research strategy is of a qualitative-empirical and constructive nature, blending various methods within the DS framework. Data collection methods include systematic, multivocal literature reviews, document analysis, and semi-structured expert interviews.

3.2. Research Methods

This section provides a comprehensive overview of the methodologies applied throughout our research by outlining the three main research methods underpinning the results of this dissertation: *systematic literature reviews*, *expert interviews*, and a *case study*. Additionally, four supplementary methods — *Observations*, *document analysis*, *unstructured interviews*, and *workshops* — enhanced our research approach. For more details, each publication from **P1** to **P5** includes a dedicated section that elaborates on the specific methods used.

3.2.1. Systematic Literature Review

In general, literature reviews systematically analyze existing literature to investigate relevant studies and their results, focusing on specific outcomes, methods, theories, or applications. The goals of these reviews may include integrating and synthesizing previous work, offering critiques, or identifying key issues and future research directions [Coo88].

A definition of a literature review that fits very well with the intended purpose of SLRs in our research is the definition by Fink. According to Fink [Fin05], a literature review is "a systematic, explicit, comprehensive, and reproducible method for identifying, evaluating, and synthesizing the existing body of completed and recorded work produced by researchers, scholars, and practitioners." In this dissertation, SLRs form the foundational methodology across all our publications (**P1-P5**), providing a rigorous base for our research.

However, due to the page limits imposed by the publishing entities, the detailed methodologies and results of our SLRs are only explicitly described in publications **P2** and **P5**. In **P1**, **P3**, and **P4**, while we rely on the results from our SLRs, we primarily focus on the primary research methods of each publication.

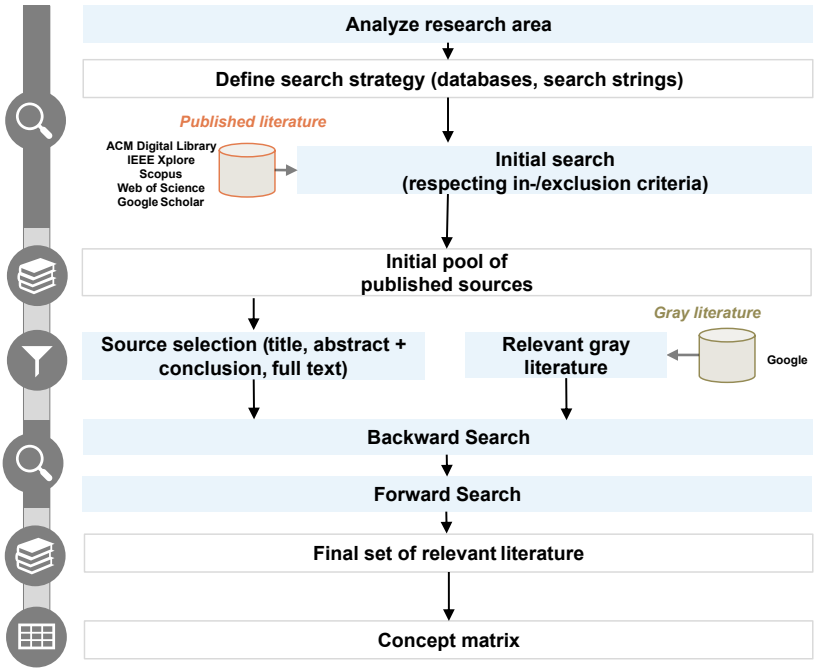

Figure 3.1.: SLR approach in P2 [NSM23]

We summarize our SLR approaches of the respective publications in the following.

In **P1**, we undertook a literature review to uncover crucial aspects of security in LSAD environments. This review helped categorize the problem domain into four key areas, detailed in the theoretical background chapter of the publication. These categories then served as the foundation for our expert interviews. Specifically, we used the best practices identified from the literature as a starting point, enabling experts to report on and discuss the approaches employed within their own organizations.

In **P2**, we present our comprehensive SLR approach based on Webster and Watson [WW02], Brocke et al. [BSN+09], and Kitchenham and Charters [KC07] with the goal to identify and analyze the current state of security compliance and governance in LSAD. Figure 3.1 shows an overview of our SLR approach in **P2**.

The SLR approach involved scanning major journals and analyzing relevant references [WW02, KC07]. A systematic search included databases like ACM Digital Library, IEEE Xplore, Scopus, Web of Science, and Google Scholar, initiated from 2001 to reflect the Agile Manifesto's [BBC+01] publication. Searches were limited to titles, abstracts, or keywords [LJE06], with source selection through relevance reviews of titles, abstracts, conclusions, and full texts. The completeness of the search was verified through backward and forward searches, ensuring no vital literature was missed [WW02, BSN+09, LJE06]. The process concluded with an extensive analysis and synthesis of all collected data, including gray literature, as proposed by Garousi et al. [GFM19].

As a preparation for the research in **P3**, we conducted a literature review, also based on the

methodology of Webster and Watson [WW02] and Brocke et al. [BSN+09]. The review involved a structured literature search and analysis. The literature search process comprised four steps, including the selection of resources, the creation of a search string with relevant keywords, and two types of searches. To analyze the relevant publications obtained from the literature search, we applied the concept-centric approach of Webster & Watson [WW02]. The resulting concept matrix yielded 27 agile security approaches, which we then used as a basis for our subsequent case study. Due to the strict page limit of the publication venue of our **P3**, we had to omit the details of the SLR conducted as a basis for **P3**, and focused primarily on the results of our case study.

In preparation for the research in **P4**, we build on the SLR conducted in **P2** and **P3** and extended it by identifying additional existing security roles and activities, building the foundation for our solution artifacts, namely the generic organizational setup of security-related roles, a team autonomy assessment model, and an adaptive collaboration model.

In **P5**, we aimed to identify influencing factors for developing secure and security-compliant applications in LSAD, as well as existing team security maturity models. We divided our SLR into four phases, which are derived from Webster and Watson's [WW02] suggestions for structuring a literature review. Initially, databases like ACM, Web of Science, and IEEE were selected, and a refined search string was developed to target relevant LSAD and security literature. The search yielded 138 publications, which were filtered down to 51 after removing duplicates and irrelevant content, with only ten qualifying for in-depth analysis based on their full texts. To broaden the literature base, both backward and forward snowballing techniques were employed, resulting in the addition of 12 more publications to capture practical insights. Finally, the gathered information was synthesized into a concept matrix to identify and document best practices for evaluating security capabilities in development teams, facilitating a structured presentation and analysis of the findings.

Finally, it is noteworthy that in **P1**, **P2**, and **P5**, we supplemented our review of academic literature with gray literature, including white papers, blog articles, and market research reports. This was achieved by extending our SLR to a so-called "multi-vocal" SLR to capture the state of the practice [GFM16], which is crucial for practitioner-oriented research areas like security in LSAD. The inclusion of gray literature, identified through backward and forward searches as well as initial screening on Google [KC07, GFM19], is justified by the relatively sparse academic literature available on this topic. This approach ensures a comprehensive understanding of current practices and challenges in the field.

3.2.2. Expert Interviews / Interview Study

Expert interviews are a crucial method for gathering qualitative data, particularly valuable in exploring complex issues within information systems research [MN07]. Such interview studies offer critical insights that either complement findings from literature reviews [RR12] or help evaluate solution artifacts [PRTV12]. Research recognizes three primary types of qualitative interviews: structured, unstructured, and semi-structured [MK11, Mye13]. Among these, semi-structured interviews are particularly prevalent in information systems research due to their flexibility [MK11]. These interviews are guided by an interview guideline that suggests questions

but allows for improvisation, enabling interviewers to delve deeper into the most relevant and interesting topics based on the interview flow [MN07, BLM09].

In all five of our research publications, we conducted interview studies as they are an effective method for collecting qualitative data across multiple organizations and industries. These studies helped us analyze the problem space—identifying challenges and best practices—and evaluate our solution artifacts. We predominantly used semi-structured interviews, which allowed for flexibility in responding to the interviewee and the ability to probe deeper into specific areas of interest, as described previously. Nevertheless, we prepared detailed interview guides with predefined questions to ensure consistency and comparability across the interviews. The interviewers were encouraged to cover all relevant topics while allowing the conversation to flow naturally, adapting the order of questions as needed to fit the dialogue, thus fostering a more engaging and productive exchange [BLM09, Mye13].

In **P1**, we detail a three-stage research methodology to explore security in LSAD, focusing on study design, data collection, and analysis. During the interview design, we conducted four preliminary interviews to discuss, improve, and verify the categorization and interview guide that we created based on our SLR. Data collection involved interviews with experts from nine companies across industries like IT, consulting, and automotive, to enhance the generalizability of our findings. With the consent of the participants, we recorded and transcribed the interviews. Data analysis was performed using the Kuckartz model [Kuc14] via the MAXQDA software [Gmb], employing a mix of deductive and inductive coding to systematically categorize and analyze the data.

In **P2**, we conducted nine semi-structured interviews with experts from seven companies on security and governance in LSAD environments, following methodologies from Myers and Newman [MN07], H.J. Rubin and I.S. Rubin [RR12], and Saldana [Sal15]. We recorded, transcribed, and employed cyclic coding techniques recommended by Saldana [Sal15], using both deductive and inductive approaches to develop accurate codes. Again, the coding process was facilitated using the MAXQDA software [Gmb], enabling effective data management and analysis. Figure 3.2 shows an overview of our **P2** interview study approach.

In **P3**, we diverged from our usual semi-structured interview format by incorporating unstructured interviews during our case study. These unstructured interviews allowed us to engage daily with a larger number of employees, facilitating a deep understanding of the case company, the identification of security approaches adopted by the case company, as well as the drivers and obstacles affecting the adoption process. Alongside this, we conducted semi-structured interviews with 10 experts to provide a more detailed evaluation of security approaches, further enriched by a survey. This mixed approach enabled comprehensive insights into the practical application of security measures within the organization.

For **P4** and **P5**, our approach focused on expert interviews to assess our solution artifacts. In **P4**, we conducted 28 interviews with 18 experts from 15 companies across diverse industries, including finance, retail and e-commerce, consulting, entertainment, and automotive. We ensured a diverse participant pool from different ASD roles and security governance to audit-related positions, encompassing roles like Product Owners, Scrum Masters, Agile Developers, Security Engineers, Architects, Business Analysts, Consultants, Information Security Leads, Security

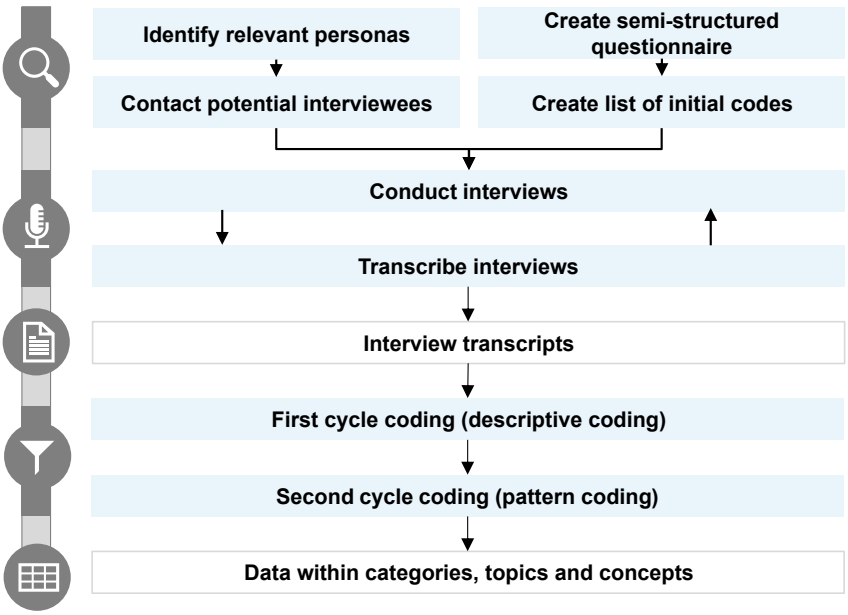

Figure 3.2.: Interview study approach in P2 [NSM23]

Managers, and Auditors. The selection was based on extensive experience at the intersection of LSAD and security. In **P5**, we carried out 12 interviews with experts averaging six years of security experience and seven years in scaling agile. The methodology for data collection and analysis in **P4** and **P5** is similar to **P1-P3**, with details available in the respective research method chapters of each publication.

3.2.3. Case Study

In **P3**, our primary research method is a case study, guided by the frameworks established by Runeson and Höst [RH09], Yin [Yin03], and Eisenhardt [Eis89]. Case studies are well-suited for exploratory and empirical investigation of phenomena within their real-life context, especially when the boundaries between the phenomenon and context are blurred and the phenomena cannot be manipulated [Yin03, BGM87]. This method is particularly effective for addressing "how" and "why" questions in contexts where empirical data is limited [Yin03].

We opted for a single-case study design, which is ideal for detailed, contextual analysis of contemporary events [Yin03], understanding the effects of specific transformations [RH09], and identifying the underlying reasons for these changes [Eis89]. This was particularly relevant as we focused on agile security practices, exploring the driving factors, obstacles, and the benefits and challenges of their implementation. Our goal was to enrich the empirical evidence in this area and provide fresh insights from the industry, thus contributing valuable new perspectives to the academic and industry discussions on agile security implementations. Our case study approach consisted of two main phases: study design, and data collection and analysis. In the beginning of our case study design, we defined the study's objectives, objects, theoretical underpinnings, and methodologies. The primary aim was to investigate the implementation of security

approaches within regulated LSAD environments. We chose one of the world's largest financial organizations for this study for two main reasons: to enrich existing research with new empirical insights on agile security in regulated sectors, and because this organization exemplifies the significant tensions between agile practices and stringent security requirements due to its recent upscale in agile methodologies under high regulatory demands in a large-scale environment. The focus of our analysis was on the organization's agile development teams and its central security department.

To ensure a comprehensive understanding and robust results, our data collection strategy involved a blend of direct and indirect methods, as suggested by Runeson and Höst [RH09] as well as Yin [Yin03]. Direct data collection included a survey, unstructured interviews, and semi-structured interviews, allowing us to gather both qualitative and quantitative data, a synergy described as highly effective by Eisenhardt [Eis89]. Indirect methods complemented this approach, involving observations and the examination of documents and archival records. The survey and the supplementary methods are described in more detail in the next paragraphs of this thesis.

3.2.4. Survey

The survey conducted in **P3** was structured according to Fowler's methodologies, focusing on sampling, question design, and data collection [Fow09]. The survey aimed to capture insights from individuals within the agile structure of the case study organization, particularly targeting SCs due to their significant involvement in security activities. Out of 62 valid responses, about one-third were from SCs.

The questionnaire consisted of 29 questions tailored to align with the study's research goals and was designed to ensure clarity and relevance, incorporating both closed and open-ended questions [Fow09], thereby collecting quantitative and qualitative data. The closed questions used a five-category ordinal scale to facilitate the analysis, with some questions including a 'not applicable' option for added flexibility. Additional qualitative feedback was requested through free-text fields. Initial feedback from a pilot group of 10 individuals led to adjustments in the wording of questions and response options to enhance the survey's reliability and construct validity [Fow09].

Data collection was executed through an anonymous online survey using TemboSocial [Inc], with the survey link distributed via email, company social network, and security channels. The results were analyzed using Excel from the exported data file, allowing us to gain a thorough understanding of the security practices and perceptions within the organization.

3.2.5. Supplementary Methods: Observations, Document Analysis, Unstructured Interviews and Workshops

To complement our primary data collection methods, we have employed additional methods. In **P3**, over the course of six months, our case study involved a thorough analysis of documents and archival records pertinent to our research, including presentations, intranet articles, and internal

reports related to security and LSAD. We also engaged in numerous meetings to observe the interactions within the security department and agile teams of the organization under study. Additionally, we held unstructured interviews and informal discussions with individuals involved in security or LSAD almost daily. As part of our research in **P5**, we conducted two three-hour workshops with secure development experts specializing in LSAD to deeply engage in discussions about our solution artifacts. These workshops allowed for a more extensive dialogue than the interviews, shaping our solution artifacts in **P5**. In particular, they influenced the selection of security maturity criteria and recommendations, as well as the structure and content of the TSMM.

Part B

CHAPTER 4

Publications

The following chapter provides fact sheets with key information of all five publications embedded in this dissertation.

The full publications can be found in Appendix A.

4.1. Publication I: Investigating the Current State of Security in Large-Scale Agile Development (P1)

Table 4.1. Fact sheet publication P1

Authors	Nägele, Sascha Watzelt, Jan-Philipp Matthes, Florian
Affiliations	Technical University of Munich, Chair of Software Engineering for Business Information Systems, Boltzmannstraße 3, D-85748 Garching, Germany
Outlet	XP 2022 23th International Conference on Agile Software Development in Agile Processes in Software Engineering and Extreme Programming
Pages	17
Status	Published
Contribution of first author	Problem definition, research design, data collection, data analysis, interpretation, reporting, writing

Abstract. Agile methods have become the established way to successfully handle changing requirements and time-to-market pressure, even in large-scale environments. Simultaneously, security has become an increasingly important concern due to more frequent and impactful incidents, stricter regulations with growing fines, and reputational damages. Despite its importance, research on how to address security in large-scale agile development is scarce. Therefore, this paper provides an empirical investigation on tackling software product security in large-scale agile environments. Based on a literature review and preliminary interviews, we identified four essential categories that impact how to handle security: (i) the structure of the agile program, (ii) security governance, (iii) adaptions of security activities to agile processes, and (iv) tool-support and automation. We conducted semi-structured interviews with nine experts from nine companies in five industries based on these categories. We performed a content-structuring qualitative analysis to reveal recurring patterns of best practices and challenges in those categories and identify differences between organizations. Among the key findings is that the analyzed organizations introduce cross-team security-focused roles collaborating with agile teams and use automation where possible. Moreover, security governance is still driven top-down, which conflicts with team autonomy in agile settings.

4.2. Publication II: The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study (P2)

Table 4.2. Fact sheet publication P2

Authors	Nägele, Sascha Schenk, Nathalie Matthes, Florian
Affiliations	Technical University of Munich, Chair of Software Engineering for Business Information Systems, Boltzmannstraße 3, D-85748 Garching, Germany
Outlet	IEEE CBI 2023 25th IEEE Conference on Business Informatics
Pages	10
Status	Published (Note: embedded in Appendix A is the accepted version due to copyright)
Contribution of first author	Problem definition, research design, data collection, data analysis, interpretation, reporting, writing

Abstract. Agile methodologies have gained popularity in software and information systems engineering due to their ability to enable rapid adaption to changing requirements and ensure business value creation in fast-paced environments. However, scaling agile to multiple teams presents challenges related to security governance and compliance. Traditional security activities struggle to keep pace with iterative agile methods. The tension between security and agility intensifies in scaled environments as governance and compliance procedures conflict with the desired autonomy of agile teams. With the increase in the number and complexity of security risks, it is imperative to better understand the current challenges and solution approaches for security governance in large-scale agile development (LSAD). To this end, we conducted a systematic literature review and an interview study involving nine industry experts. We identified 15 relevant challenges and analyzed existing LSAD frameworks concerning their solution approaches for achieving security governance and compliance. In addition, we contribute an overview of alternative solution approaches and propose five factors to balance control and autonomy to mitigate security challenges in LSAD. Our findings provide a foundation for developing well-grounded solution artifacts that address the identified challenges.

4.3. Publication III: Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry (P3)

Table 4.3. Fact sheet publication P3

Authors	Nägele, Sascha Korn, Lorena Matthes, Florian
Affiliations	Technical University of Munich, Chair of Software Engineering for Business Information Systems, Boltzmannstraße 3, D-85748 Garching, Germany
Outlet	ARES 2023 18th International Conference on Availability, Reliability and Security
Pages	12
Status	Published
Contribution of first author	Problem definition, research design, data collection, data analysis, interpretation, reporting, writing

Abstract. Agile development methods have pervaded software engineering and are increasingly applied in large projects and organizations. At the same time, security threats and restrictive legislation regarding security and privacy are steadily rising. These two trends of agile software development at scale and increasingly important security requirements are often at odds with each other. Academic literature widely acknowledges the challenges therefrom and discusses approaches to integrate these two partly conflicting trends. However, several researchers point out a need for empirical studies and evaluations of these approaches in practice. To fill this research gap, we conducted a case study in the finance industry. We identified 27 agile security approaches in academic literature. Based on these theoretical findings, we carried out observations, document analysis, and unstructured interviews to identify which approaches the case company applies. We then conducted semi-structured interviews with 10 experts and a survey with 62 participants to evaluate 14 approaches. One of the key results is that role and knowledge approaches, such as dedicated security roles and communities, are especially important in scaled agile development environments. In addition, the most beneficial security activities are easy-to-integrate, such as a security tagging system, peer security code reviews, security stories,

and threat poker.We also contribute evaluation criteria as well as drivers and obstacles for the adoption of agile security approaches that can be used for further research and practice.

4.4. Publication IV: Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development (P4)

Table 4.4. Fact sheet publication P4

Authors	Nägele, Sascha Schenk, Nathalie Fechtner, Nico Matthes, Florian
Affiliations	Technical University of Munich, Chair of Software Engineering for Business Information Systems, Boltzmannstraße 3, D-85748 Garching, Germany
Outlet	ICEIS 2024 26th International Conference on Enterprise Information Systems
Pages	12
Status	Published
Contribution of first author	Problem definition, research design, requirements engineering, data collection, data analysis, interpretation, artifact development, reporting, writing

Abstract. Companies are increasingly adopting agile methods at scale, revealing a challenge in balancing team autonomy and organizational control. To address this challenge, we propose an adaptive approach for security governance in large-scale agile software development, based on design science research and expert interviews. In total, we carried out 28 interviews with 18 experts from 15 companies. Our resulting approach includes a generic organizational setup of security-related roles, a team autonomy assessment model, and an adaptive collaboration model. The model assigns activities to roles and determines their frequency based on team autonomy, balancing the autonomy-control tension while ensuring compliance. Although framework-agnostic, we applied our approach to existing scaling agile frameworks to demonstrate its applicability. Our evaluation indicates that the approach addresses a significant problem area and provides valuable guidance for incorporating security into scaled agile environments. While the primary focus is on security governance, our insights may be transferable to other cross-cutting concerns.

4.5. Publication V: Assessing Team Security Maturity in Large-Scale Agile Development (P5)

Table 4.5. Fact sheet publication P5

Authors	Nägele, Sascha Watzelt, Jan-Philipp Matthes, Florian
Affiliations	Technical University of Munich, Chair of Software Engineering for Business Information Systems, Boltzmannstraße 3, D-85748 Garching, Germany
Outlet	HICSS 2024 57th Hawaii International Conference on System Sciences
Pages	12
Status	Published
Contribution of first author	Problem definition, research design, requirements engineering, data collection, data analysis, interpretation, artifact development, reporting, writing

Abstract. Organizations struggle to balance agile team autonomy and strict security governance in large-scale agile development environments. In particular, conventional top-down IT governance mechanisms often conflict with the desired autonomy of decentralized agile teams. Our research presents a novel approach to resolve the tension between security governance and development agility: a criteria-based security maturity assessment that enables greater autonomy for mature agile teams. Leveraging design science research, a literature review, and an interview study, we introduce two key contributions: a criteria catalog for evaluating a team's capabilities and a team security maturity model. Our expert evaluation confirms their value for systematically assessing the teams' capabilities to deliver secure and compliant applications, allowing organizations to grant more autonomy to mature teams and prioritize supporting lower-maturity teams. Future work could go beyond expert interviews and implement and evaluate the team security maturity model through a case study or experiments.

Part C

CHAPTER 5

Discussion

In this chapter, we present a summary of the core results achieved in our five main publications by answering our four RQs (Section 5.1). Subsequently, we discuss our key findings (Section 5.2), outline the potential implications of our results on research and practice (Section 5.3), and examine the limitations of our research (Section 5.4).

5.1. Summary of Results

This section summarizes our research results by answering the RQs presented in Section 1.2.

5.1.1. RQ1: Current State of Information Security and Security Governance and Compliance in LSAD

With our three first publications, **P1** [NWM22], **P2** [NSM23] and **P3** [NKM23], we answer our first RQ and thereby achieve our first contribution, which consists of revealing challenges, drivers, recurring patterns and solution approaches for security and LSAD integration.

Due to the broad scope of our first RQ, we use the sub-questions presented in Section 1.2 to structure our results.

Research question 1 (RQ1)

What is the current state of information security and security governance and compliance in (large-scale) agile development?

RQ1.1: *What are recurring challenges of security in LSAD?*

In **P1**, we identified two core challenges recurringly appearing in the industry in multiple studied case companies.

The first challenge is the lack of personnel with sufficient experience in both security (governance) and ASD. In many cases, companies can access resources that have expert knowledge and many years of experience in one of those areas, but rarely both. Due to the different focuses, style of working and cultural mindset, this leads to challenges. The scaled agile environment amplifies the problem because centralized security teams have frequent contact with agile teams due to short development cycles. Also, the expected response times of security experts to inquiries of agile teams are lower, resulting in a higher pressure on central security experts and possible frictions and delays in the development process.

The second challenge is the conflict between security governance and team autonomy when coordinating many teams. Teams should work as autonomously as possible, yet security policies and standards must be defined and managed. Scaling makes it challenging to monitor and control, as it is no longer possible to "look over the shoulders of the developers", as one expert stated.

Since identifying and analyzing challenges was only a small sidequest in the broader endeavor to illuminate the state of security in LSAD in **P1**, we focused our research in **P2** on a much more thorough investigation of challenges. In total, we identified fifteen relevant and recurring challenges that we categorized in three groups: First, challenges that are specific to security in LSAD environments. Second, challenges originally already identified and reported in small-scale agile development, but still relevant and thereby transferable to LSAD. Third and finally, we also identified challenges that appear in LSAD, but also apply and are relevant to our security context.

An overview of all these challenges is shown in Figure 5.1.

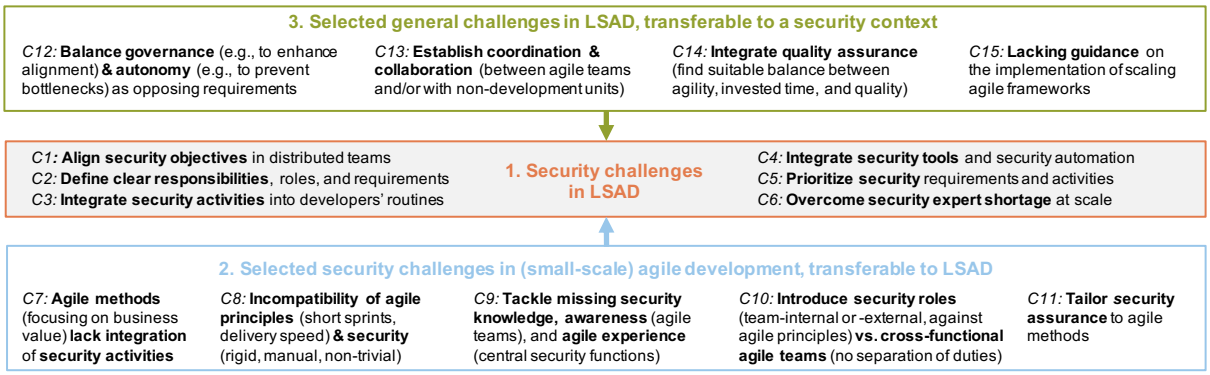

Figure 5.1.: Overview of identified security challenges in LSAD based on P2 [NSM23]

RQ1.2: *How do existing solution approaches and best practices for integrating security (gover-*

nance and compliance) in LSAD environments look like?

With the results of **P1**, we already provide a detailed look into existing solution approaches and best practices for integrating security in LSAD environments. Specifically, we identify and analyze four areas that these solution approaches and best practices can be categorized in. In the following, we briefly describe the results of our research on these areas.

In addition to the results that originate from our review of existing literature, we contribute our own results in these areas as well, also included in the following explanation of each category.

Table 5.1 shows an overview of the 17 solution approaches and recurring patterns we identified in the analyzed companies and the average state of maturity of that approach within these areas, as estimated by the interviewees.

Structure of the agile program.

Organizational structures, including the roles and their interactions, are crucial for enhancing security within LSAD. For example, Poller et al. [PKT+17] highlight the critical importance of understanding roles and their interactions as a foundation for effective security governance in LSAD environments. This is further supported by Alsaqaf et al. [ADW17], who note the introduction of specific roles, such as security architects, to meet emerging quality and security demands, thereby indicating a significant gap in empirical research concerning these roles.

The work of Newton et al. [NAD20] and Rindell et al. [RHL18] reveals the presence of security-focused communities and internal groups dedicated to activities like security reviews, showcasing proactive approaches to embedding security within these agile settings. Moreover, the observations by Steghöfer et al. [SKHW19] and Dännart et al. [DCB19] that LSAD frameworks inherently lack security compliance highlight the necessity for adaptations, particularly the integration of specialized security roles.

Oyetoyan et al. [OCJ16] describe the role of security expert groups in supporting adherence to security standards and organizing security audits, underlining the importance of expert guidance in maintaining security protocols. Additionally, reports and documentation from leading software companies, including SAP [SAP20], Microsoft [Mic16], and Google [Clo24], illustrate the application of dedicated security roles within industry practices, despite a lack of detailed task descriptions.

Collectively, these findings emphasize the indispensable role of a well-defined agile program structure in addressing and enhancing security measures within LSAD, especially the specification and integration of security roles.

In regards to the organizational structure, we identified four important roles relevant to our research: central security teams, team-internal and external security roles as well as communities and their members.

We found that most organizations employ centralized security teams consisting of specialized roles like penetration testers, security analysts, or information security officers to collaborate with agile programs. These central teams are responsible for setting security criteria, conducting security verifications, compliance checks, risk analyses, and reviews (including code reviews and

Table 5.1. Overview of recurring best practices based on P1 [NWM22]

	01	02	03	04	05	06	07	08	09
Integration of security activities									
Security self-assesment	●	●	◑	○	◑	●	●	●	◕
Bug bounty		◑		◑	●	○	●	○	●
Threat modeling	◑	●	●	●	◕	◑	◑	●	◕
Penetration testing	●	●	●	●	●	●	●	●	●
Security audits		●	●	●		●	●	◑	●
Security code review	◑	◑	●	○	●	●	○	●	◑
Tool-support and automation									
DevSecOps pipeline	◕	●	●	◕	●	◑	◑	●	●
Static code analysis	◑	●	●	●	◑	○	◑	●	●
Vulnerability scanning	◑		●	◕	●	●	◕	●	◑
Dependency checks	●	◑	●	◕		○	◑	○	◑
Security governance									
Bottom-up	◑	○	○	○	◑	○	◑	◔	◔
Top-down	●	●	●	●	●	●	●	●	●
Reusable components		●	●		◑	○	◕	◕	◑
Organizational structure									
Security champion	◑	●	○	○	●	○	●	◕	◑
Security engineers or architects	◕	●	◕	●	◑	●	●	●	●
Central security teams	●	●	◕	◕	●	◑	●	○	◕
Communities of practice	◕	●		◑	◕	●	●	●	●

none: ○ | rare or planned: ◔ | partial: ◑ | frequent: ◕ | complete: ●
no classification possible: *empty*

penetration tests). Some tasks, such as threat modeling, are done jointly with development teams, facilitating knowledge transfer and potentially enabling these teams to independently handle security tasks in the future. Central teams may also audit and approve changes before production deployment, especially for critical software, but face scalability issues when working with multiple agile teams, leading to bottlenecks.

To address this, organizations introduce security-focused roles within agile teams to increase

security capabilities and autonomy. Team-internal roles, like SCs, are developers with additional security training, increasing security awareness within their teams while still holding the entire team accountable for the application's security. Meanwhile, team-external roles, such as SEs or consultants, support multiple teams by providing security expertise and acting as a bridge to central security departments. They may conduct threat modeling workshops and ensure the transfer of knowledge on laws, policies, and best practices.

Cross-team collaboration on security is enhanced through regular meetings, training, communities of practice, guilds, or chapters, with the aim of sharing security knowledge. Organizations also utilize corporate social networks and wikis for documentation and expert search, though the effectiveness of knowledge sharing is challenged by the complexity and specificity of documentation. Practical code examples are highlighted as particularly beneficial for developers in understanding security concepts.

Figure 5.2 shows a generalized organizational setup based on these empirical results.

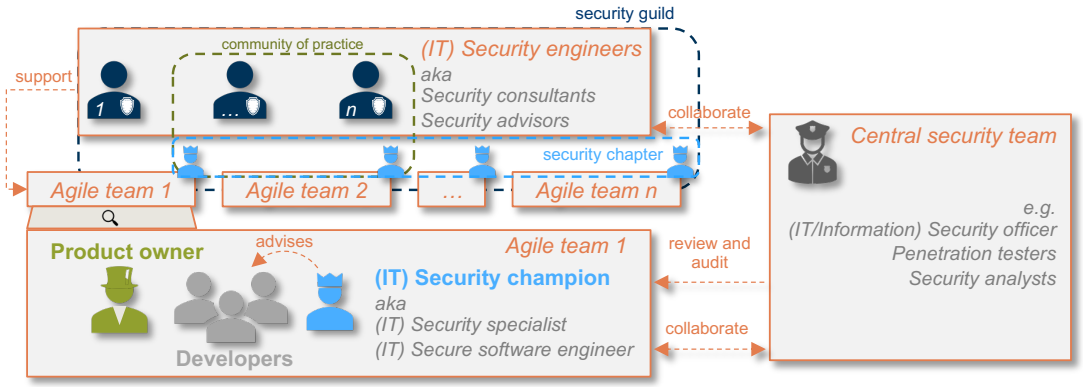

Figure 5.2.: Generalized structure of security integration in LSAD based on P1 [NWM22]

We further investigated these findings on organizational structure in a case study in **P3**. The observed structure in the case company is as follows. There is a central security function distinct from the software development teams, with specialized roles in governance, compliance, and technical operations. The governance and compliance roles are led by a security officer responsible for policy definition and audits, whereas a technical team handles aspects like security testing and incident response. In addition to the central team, team-internal SCs lead the security efforts of cross-functional agile development teams, while they are supported by SEs who are not directly part of but support multiple teams.

Additionally, the organization has introduced so-called guilds, mirroring the Spotify model [KI12], to foster collaboration between agile teams and governance functions on key focus areas such as security. These guilds facilitate lean governance by promoting knowledge sharing and empowering teams to adhere to and influence the evolution of organizational standards and guidelines.

Security governance. Security governance, a branch of IT governance, traditionally employs a top-down approach to control [Ins06]. However, the relevance of adapting governance to fit agile and lean environments, despite limited empirical research, is increasingly recognized [VRC19].

Industry literature and agile frameworks [AL19, SA24b] are now more frequently discussing lean governance, indicating a shift towards practices that better integrate governance with agility.

Research by Horlach et al. [HBSD18] indicates that traditional governance structures may obstruct the autonomy of agile teams in LSAD, suggesting a move towards leaner governance models. Ambler [Amb08] and Vejseli et al. [VRC20] further support this shift, highlighting that agile IT governance can enhance business-IT alignment and improve enterprise performance, similarly to traditional governance but with added agility.

Agile governance prioritizes empowering development teams by encouraging collaborative and supportive practices, moving away from rigid control towards promoting autonomy and self-organization [Amb08, AL19, LKRM16]. This approach not only fosters greater business agility but also frames security governance as an essential component, balancing organizational directives with team independence.

The companies we studied in **P1** predominantly utilize a top-down approach for security governance, with centralized teams establishing standards based on regulations, international guidelines, and best practices. There is variance in how much development teams can influence these security governance standards. Some companies strictly limit teams' input, emphasizing product development focus, while others allow for a degree of bottom-up contribution, enabling teams to propose adjustments to internal standards when justified. A highlighted effective strategy for security governance in LSAD involves providing standardized, security-focused components for reuse by development teams, such as identity management, input validation, data encryption, and secure communication protocols. These components not only facilitate application security verification but also face challenges like outdated documentation, usage uncertainties, and a general lack of awareness.

Security activities.

In the context of our research, security activities are defined as practices that either directly or indirectly contribute to enhancing software security. A prime example of such an activity is threat modeling, which plays a crucial role in security risk analysis [MS16] by aiding in the identification of potential security threats and the determination of mitigating measures [Sho14]. Other notable activities include penetration testing [ASM05] and code reviews [SSC+18], which are widely recognized for their contributions to software security.

The feasibility and necessity of integrating security activities within agile development processes have been well-documented, with a consensus among scholars on this matter [MAR+20, NAD20]. Beznosov and Kruchten [BK04] suggest that the integration of security practices into agile methodologies should be tailored based on how well these practices align with agile principles. Keramati and Mirian-Hosseinabadi further elaborate on this integration, noting that it involves a trade-off between the potential decrease in agility and the advantages of developing more secure software systems [KM08]. This analysis leads us to categorize 'security activities' as a key theme for our interviews, underlining their significance in the development of secure agile software.

Our interviews with experts in **P1** revealed a range of activities deemed crucial for enhancing software security.

Common strategies include code reviews and pair programming for quality assurance, with an emphasis on incorporating security aspects despite challenges like time consumption and potential oversight of security issues. Regular penetration tests and bug bounty programs are crucial for identifying vulnerabilities, although penetration tests face limitations related to their discontinuous nature and the feasibility of conducting them for minor product increments. Security reviews and audits ensure compliance with standards, adopting a flexible approach to frequency and scope based on the application's criticality.

Threat modeling is valued for its fit with iterative development, allowing for targeted security enhancements based on incremental changes. Security self-assessments help teams gauge compliance and security relevance, promoting a culture of security awareness and accountability, albeit with some finding the process overly time-consuming. The practice of security risk management, where product owners document and take responsibility for acceptable risks, facilitates the proactive identification and documentation of potential security issues. Meanwhile, the challenge of maintaining comprehensive security documentation in fast-paced agile environments is mitigated by using tools for automatic generation and incremental updates, ensuring documentation keeps pace with development.

These practices underscore the nuanced balance between rigorous security measures and the agility of development processes, highlighting tailored approaches as essential for effective security in LSAD.

Tool-support and automation.

In the context of integrating security into agile methodologies, Barbosa and Sampaio [BS15] highlight the challenge of balancing the need for quick, cost-effective software development with the time and financial costs of agile security practices. This challenge underscores the importance of automating labor-intensive tasks to minimize the conflict between security measures and rapid deployment cycles. The evolution of DevSecOps from a conceptual buzzword to a key strategy in modern development underscores this shift towards incorporating security practices seamlessly into development workflows through automation tools [MC17].

Automation, particularly in repetitive tasks such as security code reviews, is emphasized as crucial for maintaining agile development's speed without compromising security [JSMB13, DWA22]. Techniques like SAST and DAST exemplify this automation, aiming to streamline the integration of security into the development process. Given the importance of minimizing manual effort and enhancing the smooth incorporation of security practices in larger-scale projects, 'tool-support and automation' emerged as a critical category for our research in **P1**, highlighting the essential role of automation in reconciling security with agile development speed and efficiency.

Based on our results from **P1**, all interviewed companies employ DevSecOps pipelines during the build and deployment phases of their applications, incorporating both SAST and DAST, alongside the implementation of metrics and quality gates, to enhance application security.

SAST is widely used for identifying potential security vulnerabilities in code, albeit with challenges such as managing false positives and outdated dependencies. DAST, though less mature,

focuses on automating parts of manual security assessments like penetration testing and vulnerability scanning to identify infrastructure weaknesses.

The incorporation of automation tools within these pipelines generates critical security metrics, aiding in the determination of whether applications meet predefined security thresholds for production release. Despite the reliance on automation for efficiency, there is a consensus on the importance of not solely depending on automated tools due to their limitations. The evolving role of machine learning in security testing suggests a future where the capabilities of automation could significantly expand, potentially reducing the need for manual testing.

RQ1.3: *What security approaches are being used in a specific LSAD case study environment?*

To further investigate part of the categories of the previous subquestion RQ1.2 in more depth, we conducted a case study over the period of six months with an exemplary organization that has a LSAD environment. One of our case study goals was to investigate what agile security approaches are already adopted by the case company. The results are summarized in the following.

We categorize the identified approaches into role and knowledge approaches and methodological approaches in the case company.

Adopted roles and knowledge approaches.

We identified four agile security approaches that revolve around roles and knowledge integration, all driven by the establishment of a security community which is called a "security guild". This guild is central to embedding security within agile methodologies, ensuring that security becomes a routine and integral activity across teams. It comprises three pivotal roles: a security expert embedded in each team (called SC), a team-external role (called SE), and a security officer, coordinating broad technical security issues and enhancing security competence among developers to foster independent security decision-making.

To facilitate this, the company has implemented several mechanisms for exchanging security knowledge. For instance, SEs hold monthly sessions to discuss current security topics and challenges, and they frequently engage with SCs to disseminate this knowledge to the development teams. This exchange is supported by various channels, including company-wide events, internal social networks, and dedicated security channels in internal messengers where topics like cloud security, risk management, and system hardening are discussed.

The team-external security role, filled by SEs, involves guiding agile teams on security practices, from guideline adherence to risk analysis and the implementation of security measures. SEs play a crucial role in mentoring team-internal security experts and in developing operational security policies that translate strategic requirements into actionable practices.

Furthermore, over 55 agile team members serve as SCs, taking on responsibilities such as creating security documentation, organizing security measures, and assisting in risk analysis. While SCs are deeply involved in monitoring and enhancing the security posture of their products, they do not assume the full responsibilities of a security architect.

Lastly, the security department conducts regular training for all employees, with specialized

sessions for product owners, SCs, and developers to boost their security skills and awareness. These training initiatives cover a range of topics from security fundamentals to secure coding, and are made accessible through recordings and short informational videos on the company's internal network, ensuring ongoing education and engagement in security best practices.

Adopted methodology approaches.

In our study, we observed seven different security methodology approaches within the case study company, each varying in maturity and implementation stages. The company uses an extended version of security stories to check user stories for security relevance, which helps in organizing and raising awareness of security issues within agile teams. Security criteria are also integrated into the definition of done for user stories, often with the guidance of security officers or SEs.

Pair programming is a common practice in the company's agile transformation initiative, and incorporating a SC or SE into this process is seen as beneficial for enhancing security focus, although this aspect was not fully realized at the time of the study. Regular security meetings between SEs and SCs adapt to the needs of the teams, focusing on urgent security concerns like security architecture or the outcomes of penetration tests. These meetings often culminate in a final security review necessary for approving critical software releases.

Regarding documentation, development teams maintain various security-relevant artifacts, such as architecture diagrams and risk assessments, which sometimes require reviews by security experts. The security department aids this process by providing checklists, templates, and a central tool for compliant storage.

Security testing varies across teams and is influenced by the criticality of the product. Penetration testing is mandatory for new applications and significant updates, executed either in-house or by external contractors. Bug bounty programs offer another possibility for continuous security testing, valuable for products with frequent releases. Teams are supported by SEs in tracking and rectifying vulnerabilities, with re-tests conducted to confirm the effectiveness of the solutions.

Lastly, security code review practices include the use of tools for static code analysis to identify security vulnerabilities, alongside other code issues. While there is no set frequency for these reviews, decisions are made based on team-specific needs, occasionally supplemented by manual peer reviews led by SEs and SCs.

RQ1.4: *What agile security integration approaches are proposed by existing scaling agile frameworks and related literature?*

Initial results from our literature reviews and expert interviews indicated that established scaling agile frameworks lack suficient guidance on security governance and compliance integration.

However, to prevent our research from, proverbially speaking, reinventing the wheel, in case some scaling agile frameworks already include answers to our RQs, we aimed to identify solution approaches in established scaling agile frameworks from research and practice.

Security governance and compliance in LSAD frameworks.

In **P2**, we conducted an analysis of existing LSAD frameworks by searching for any content related to the keywords "governance", "compliance", or "secur*" in the scaling agile framework publications.

Table 5.2 shows the coverage results of the frameworks that cover at least one of the terms in a minimal way. The categorization ranges from "nothing (relevant) mentioned" (-) to "low", "medium", and "high". We marked the coverage as "low" if a framework mentions a keyword but only partly addresses our research area. "Medium" coverage signals broader information with more detailed information on relevant aspects, whereas "high" coverage includes concrete guidance for organizations on the respective topics. We omitted frameworks from the table that did not provide any coverage of the relevant keywords at the time of our analysis, such as LeSS [LV16], Nexus [SS], and the Spotify Model [KI12].

Table 5.2. Framework coverage analysis results based on P2 [NSM23]

	secur*	**compliance**	**governance**
Scrum@Scale	-	low	-
Nexus	-	-	-
LeSS	-	-	-
Spotify model	-	-	-
SAFe	medium	medium	medium
DA	medium	medium	medium

Scrum@Scale primarily suggests the need for an independent compliance department to facilitate collaboration with non-development functions, but it falls short in offering detailed strategies for such collaboration, leading to its classification as having low coverage. On the other hand, SAFe promotes business agility, integrating security and compliance within its core values and advocating for automation and lean governance to achieve continuous compliance. However, it lacks explicit instructions on operationalizing these principles, especially in prioritizing non-functional requirements and detailing collaboration of agile teams with central departments.

DA emphasizes the role of security and risk management, recommending the inclusion of SEs and specialists in teams, treating security as a non-functional requirement, and advocating for the use of security testing tools and threat modeling. It proposes governance as an enabling function and suggests practical tools for supporting compliance. Despite these provisions, DA does not offer clear guidance on integrating these practices within agile methodologies or defining the distribution of responsibilities, thus receiving a medium coverage rating.

Overall, while SAFe and DA recognize the importance of security, governance, and compliance in agile environments, they stop short of providing detailed implementation guidance.

As explained in **P2**, these analysis results point to a gap in current agile frameworks regarding the integration of these critical aspects, highlighting areas for future development and research to provide clearer guidance for organizations to follow.

Agile security governance and compliance in related literature.

In our SLR in **P2**, we identified three categories of security integration approaches that are particularly relevant to our research: LSAD approaches focusing on security, small-scale agile approaches, and SSE practices adapted to agile methods. We summarize the results of **P2** in those three categories in the following.

Starting with LSAD, Moyon et al. [MBK+18, MMBK21] developed the S2C-SAFe model, which tackles the conflict between security compliance and LSAD by enhancing the SAFe framework with additional security roles, activities, and artifacts. This includes security requirements in the backlog derived from threat modeling and the inclusion of secure coding standards in the Definition of Done (DoD). Poth et al. [PJR20] focus on compliance in regulated environments, advocating for giving agile teams as much autonomy as possible without compromising product delivery and compliance. Petit and Marnewick [PM19] propose the "Earn Your Wings" approach, which assigns levels of autonomy based on team maturity and past performance, suggesting that increased autonomy reduces the need for workarounds and accelerates release processes.

In the realm of small-scale agile approaches, Boldt et al. [BJBC17] emphasize the necessity of systematically addressing security throughout the development process and integrating security roles to maintain security costs at reasonable levels. Hanssen et al. [HHS+16] and Fitzgerald et al. [FSOO13] introduce SafeScrum and R-Scrum, designed for safety-critical environments but also applicable to security due to the overlapping concerns.

Lastly, adapted SSE practices include the Microsoft Security Development Lifecycle [Mic], which allows for flexibility in meeting security requirements across different sprints or releases by distributing activities over time. The SAMM Agile model [OV] by the OWASP Foundation details how to incorporate security activities into sprints, enhance collaboration between security teams and developers, and promote the use of automated testing. This model also supports the introduction of a SC to foster team autonomy.

Together, these results provide a comprehensive view of current strategies for integrating security into agile processes, ranging from large-scale frameworks to tailored practices for smaller teams. We used these insights and experiences to inspire and influence the creation of our own solution approaches presented mainly in **P4** and **P5**.

RQ1.5: *What are the drivers of adopting security activities in LSAD?*

Our research in **P3** also provides some insights into the drivers of adopting security activities, meaning the aspects that propel better adoption of security practices in LSAD.

Table 5.3. Drivers and obstacles in the adoption process of agile security approaches based on expert interviews as part of the case study in P3 [NKM23]

Drivers	No. of interviewee references
Instrinsic motivation	5
Ease of approach	4
Clear benefit of approach	4
Similarity to known practices	2
Regulation or internal guidelines	2
Obstacles	**No. of interviewee references**
Lack of security awareness & knowledge	7
Need for more resources	6
Focus on functionality	5
Status of security in the organization & teams	4
Top-down vs. bottom-up driven adoption	4
Lack of agile security support	2

Table 5.3 shows an overview of all driving factors and obstacles identified in our case study research in **P3**.

The primary driving factor for adopting agile security approaches, as highlighted in expert interviews as part of **P3**, is the intrinsic motivation of software developers. This motivation often stems from a personal interest in security or the enjoyment derived from applying agile security methods. It was noted that teams usually adopt these practices on their own initiative rather than at the behest of organizational directives or product owner demands. The willingness to secure products and protect customers also plays a crucial role, although the level of intrinsic motivation varies significantly across teams.

Ease of adoption is another critical factor. Approaches that are simple to implement and do not require a substantial increase in knowledge are more readily embraced. Highlighting the tangible benefits of these security approaches to agile teams is also essential for fostering adoption. The clear demonstration of added value and understanding the purpose behind these practices further encourage teams to integrate them into their workflows. This aligns with the notion that the decision to adopt certain security practices in agile environments is influenced by the perceived cost-benefit ratio, reinforcing the importance of emphasizing practical advantages and manageable effort in promoting agile security practices.

RQ1.6: *What obstacles occur when adopting security activities in LSAD?*

Based on our **P3** findings, the adoption of agile security approaches faces several obstacles, with a primary challenge being the lack of security awareness and knowledge among developers, product owners, customers, and sponsors. This gap is often attributed to insufficient experience with security practices and the underrepresentation of agile security methods. Additionally, the presence of team members with security expertise is crucial for the effective use of these methods, as supported by both the interviews and scientific research.

Resource constraints, specifically limited budgets and time, also hinder the adoption process. Agile teams, often stretched by functional requirements, may deprioritize security improvements due to these constraints. The initial secondary status of security in agile transformations and previous negative experiences with security measures, such as release delays, can further discourage teams from embracing security practices.

The focus on functionality over security, the lack of documentation and support for integrating security into agile processes, and the absence of a clearly responsible entity for security in self-organizing teams have been identified as additional barriers. However, initiatives like the introduction of a security community have shown success in overcoming these challenges by providing needed guidance and integrating security more seamlessly into agile environments.

Effective adoption requires a balanced approach that includes both top-down commitment from management and bottom-up participation from agile teams. Encouraging involvement in the selection and evaluation of security practices and simplifying the adoption process through familiar terminology and examples are key strategies for overcoming these obstacles and enhancing the integration of security within ASD.

Summary. We summarized the core components of our answer to RQ1, which provides a holistic overview of the current state of security in LSAD. It includes identified challenges of security in LSAD, existing solution approaches, and best practices regarding four categories, which are (i) structure of the agile program, (ii) security governance, (iii) security activities, and (iv) tool-support and automation. It also presents the observed and applied security approaches from a LSAD case study environment, including roles and knowledge approaches as well as methodology approaches. Further, it identifies solution approaches from existing scaling agile frameworks and related literature. Our answer to RQ1 concludes by presenting drivers and obstacles when adopting security activities in LSAD.

5.1.2. RQ2: Roles, Activities, and Key Influencing Factors to Balance the Autonomy and Control Tension and Improve the Security in LSAD Integration

Research question 2 (RQ2)

What are security-related roles, activities, and key influencing factors to balance the autonomy and control tension and improve the security in LSAD integration?

Recommended security-related roles in LSAD.

In **P1** through **P3**, we developed a comprehensive understanding of the roles relevant to secu-

rity in LSAD. Building upon these insights, **P4** uses DS and additional expert interviews to propose an organizational structure that outlines specific roles and responsibilities for effective security integration within LSAD. This organizational setup further details, extends, verifies, and evaluates our initial findings regarding organizational structures presented in **P1**, **P2**, and **P3**, as summarized in Subsection 5.1.1 of this thesis.

Figure 5.3 depicts the resulting solution artifact summarizing our insights into the relevant roles and their responsibilities.

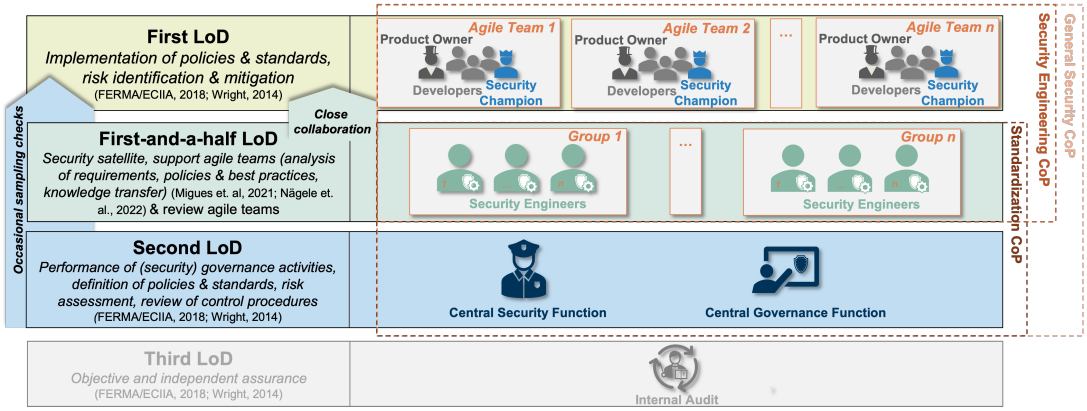

Figure 5.3.: Generic organizational structure of security-related roles based on P4 [NSFM24]

Our proposal in **P4** draws from the LoD model, adapting it to fit the agile and security landscape. A notable innovation is the introduction of the "first-and-a-half LoD", occupied by SEs who assume a dual role: They support agile teams directly as part of the first LoD and also conduct reviews as part of the second LoD, focusing on teams and projects outside their immediate support domain to maintain objectivity and prevent conflicts of interest. This model underscores the SEs' expertise in software and security engineering, enabling them to aid multiple teams in enhancing security measures and automation.

Agile teams, forming the first LoD, consist of a product owner and developers, with the SC role being an integral part of this layer. The SC is a developer with an added focus on security, acting as the main security contact within the team and towards external stakeholders, enriching the team's security posture without introducing a separate entity. This setup builds on existing literature and practical observations that outline the responsibilities and impact of SEs and SCs in leading and reinforcing the security efforts of their teams.

The second LoD is responsible for setting security policies and standards and ensuring compliance of the results of the first LoD with these requirements. It supports the first LoD by providing security tools, automated CI/CD pipelines, and training, facilitating a security-minded development process without regular direct involvement in team-specific activities.

Finally, the third LoD offers a sporadic evaluation of the security practices implemented by the preceding defensive lines, primarily through internal audits conducted independently from the development workflow. This external perspective ensures a comprehensive assessment of the

security measures across the organization, reinforcing the overall security posture within agile development settings and fulfilling regulatory and auditing requirements.

Evaluation of security activities in LSAD.

In **P1** and as part of our answer to RQ1 in Subsection 5.1.1, we already provide first insights and results on the topic of security activities in LSAD, which is an important aspect when aiming for better security integration and balancing the autonomy control tension.

In pursuing to answer our second RQ, we provide more detailed insights regarding security activities in LSAD in our third publication, especially addressing the question of what the benefits and drawbacks of these activities are, and how these activities can be evaluated in terms of their suitability in LSAD environments.

Specifically, we gained a list of 27 agile security approaches identified through a SLR and used these as a basis for our case study. We filtered the identified approaches from the literature by eliminating the ones with only very few mentions and evaluated the remaining approaches. This resulted in the evaluation of 14 agile security approaches, which we will summarize in the following.

For the purpose of the evaluation, we also developed 12 generalized evaluation criteria for agile security approaches. These can be used by other researchers or practitioners to evaluate other security approaches not included in our study, or the same approaches in other contexts or organizations.

Table 5.4 shows the evaluation criteria we established as a preparation for our own evaluations within the case study in **P4**. We used the results of a literature review we conducted prior to our evaluation to select these criteria. They enabled us to systematically analyze the compatibility of the security activities and approaches within the regulated LSAD environment of the examined organization.

Table 5.4. Evaluation criteria for agile security approaches based on P3 [NKM23]

Agile criteria [GAJ14, KM08, SBK05]	
Iterative	For example, in each iteration or regularly in other time intervals.
Adaptable	Adaptive and flexible to changing requirements.
Interactive	Focused on people and their interaction.
Lean	Accelerating software delivery, maximizing resource utilization or reducing *waste*, such as unnecessary documentation effort.
Simple	Easy to understand and easy to use.
Motivating	For example, by using fun components.
Large-scale criteria [DPL16, DMFS18]	
Autonomy	Increasing the autonomy of agile teams, for example, through increased security knowledge.
Collaboration	Improving cross-team collaboration and coordination between agile teams.
Knowledge exchange	Enhancing security knowledge exchange between agile teams.
Technical consistency	Promoting technical consistency across agile teams and the reduction of technical debt such as risk acceptances.
Regulatory criteria [FSOO13, HN18]	
Traceability	Fostering the traceability of regulatory and company-internal security requirements.
Compliance	Facilitating compliance with regulatory and company-internal security guidelines.

We then applied these criteria to evaluate security approaches with regard to their suitability for achieving security in LSAD. The results summary is shown in Table 5.5. Interviewees evaluated the approaches against each criterion, rating them as positive, neutral, or negative. The aggregate scores represent the arithmetic mean of all ratings provided by the interviewed experts. These scores range from minus one to one on a scale, where one signifies the most positive evaluation and minus one indicates the most negative assessment.

We will now briefly summarize the results shown in Table 5.5. More detailed results and explanations can be found in **P3**.

The concept of a *security backlog* receives mixed reviews, with some experts criticizing the additional effort and potential neglect, while others see its value in explicitly capturing security needs.

Security Criteria in the DoD is acknowledged for embedding security into the development process, though its implementation faces challenges in specificity and verification.

Agile risk analysis through a *security tagging system* is mostly praised for its transparency and ease of integration into team workflows, promoting ongoing security vigilance.

In contrast, *protection poker,* although not seen as suitable for all LSAD environments, is noted for its engaging and awareness-raising aspects.

Approaches like *attack trees* and *countermeasure graphs* are viewed as somewhat suitable, offering clarity on security threats but criticized for their abstract nature and the effort involved.

Threat poker is appreciated for its interactive, fun aspects, potentially enhancing team focus on security, despite being time-consuming.

Security meetings, sprints, and spikes are discussed, with opinions divided over the practicality and acceptance of dedicating time exclusively to security within agile cycles.

Pair penetration testing is valued for its educational potential, though its feasibility and effectiveness depend on external testers' willingness to collaborate and the nature of the testing process.

Peer security code reviews are broadly supported for their relevance and ability to directly impact software security, encouraging knowledge sharing and fostering a collaborative culture. Despite concerns over the time required and the need for a supportive feedback culture, this approach is highlighted for its long-term quality benefits and motivational aspects.

In summary, the findings highlight the importance of practicality, team involvement, and the balancing act between thorough security measures and iterative agile development.

Table 5.5. Evaluation results of agile security methodology approaches per evaluation criteria based on P3 [NKM23]

Agile criteria	Security stories	Criteria in DoD	Tagging system	Attack tree/ CM graph	Threat Poker	Pair pen. Testing	Peer code Reviews
Iterative	1	1	1	1	1	1	1
Adaptable	0.7	-0.5	1	0.5	1	0	0.8
Interactive	0.8	-0.5	0	1	1	0.8	0.8
Lean	0.3	1	0	-1	0	0.5	0.4
Simple	0.3	1	1	-1	1	-0.3	0.4
Motivating	0.2	-0.5	-0.3	0.5	1	1	0.6
Large-scale criteria							
Autonomy	0.3	0	0.7	-1	0	-0.3	0.4
Collaboration	0.7	0.5	1	0.5	1	0.8	0.6
Knowledge Exchange	0.8	-0.5	0	0.5	1	0.3	0.6
Technical Consistency	0.5	1	0.3	0	0	0.8	0.2
Regulatory criteria							
Traceability	0.2	0.5	1	-0.5	0.5	-0.8	-0.2
Compliance	1	0.5	0.7	0.5	1	0.5	0.2

We also conducted a survey to evaluate security approaches from the roles and knowledge category. An excerpt of the survey results is shown in Figure 5.4.

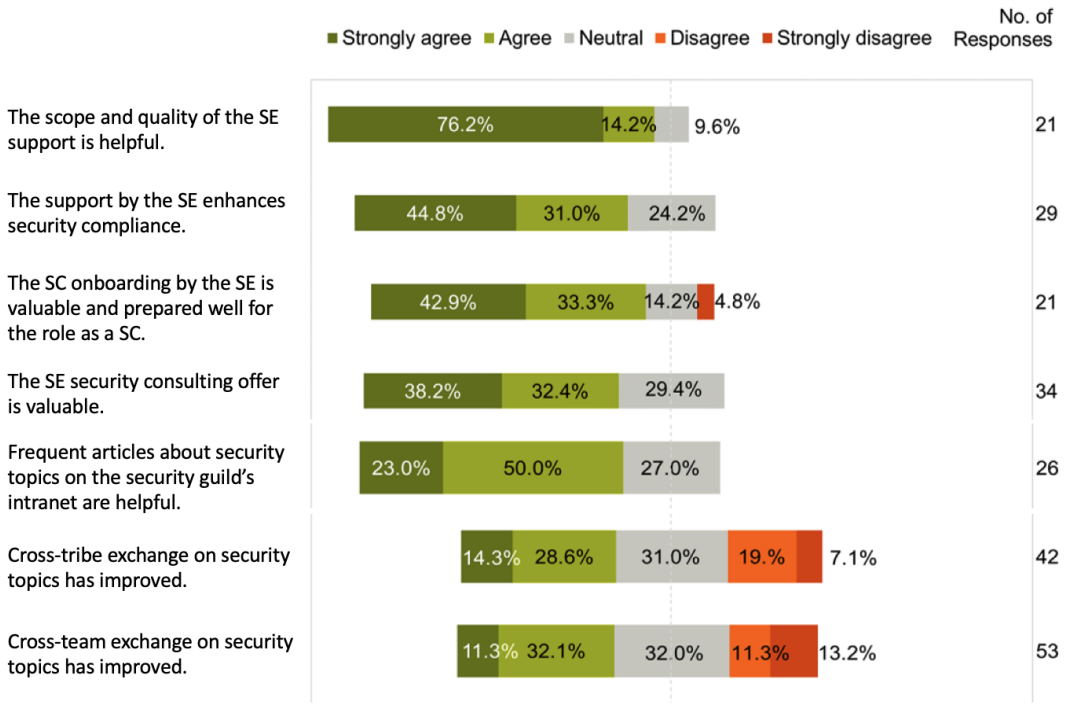

Figure 5.4.: Excerpt from the survey results on the evaluation of security role and knowledge approaches in P3 [NKM23]

In our survey in the case company with the goal of evaluating role and knowledge approaches to integrating security practices within their LSAD environment, the security guild community, formed of agile team members and central security experts, emerges as a pivotal component. The majority of respondents, 32 out of 53, value the guild's contributions to their agile tribes and teams, highlighting its role in enhancing policy compliance and security levels. Approximately 72% believe in the guild's long-term impact on maintaining appropriate security standards, with 40 respondents noting an increase in cross-team knowledge exchange, enhancing problem-solving and transparency.

Despite these positive outcomes, our analysis revealed room for improvement. About 26% of participants have not noticed enhanced security exchange across tribes, and a similar percentage sees limited progress in team interactions. Additionally, not all respondents are fully aware of the guild's offerings, though those who are engaged find the information valuable.

The support from SEs is rated highly by 70% of respondents for providing clear, competent security guidance. SCs, in particular, appreciate the onboarding and mentoring from SEs, although some call for a more precise definition of their role. The majority affirm that SE support has improved their adherence to security guidelines and mitigated issues arising from ambiguous security policies.

The introduction of the SC role is viewed as beneficial, with SCs keen on contributing to the guild and emphasizing the need for practical training to effectively support their teams. While most SCs find the training valuable, there's a desire for more specialized knowledge, such as agile security risk assessment techniques.

Security training facilitated by the guild is well-received, with various formats catering to different learning preferences. A significant majority of agile team members who engaged with the training report an increase in their security expertise, with a strong intention to continue leveraging these resources. Monthly information and training meetings by the guild are also seen as valuable, indicating a high interest among agile developers in enhancing their security knowledge through these initiatives.

To summarize, our case study results demonstrate the value of employing security roles that fit well into agile settings at scale, such as the SE and SC, as well as communities and training and exchange formats fitted to agile environments.

Factors influencing security responsibility.

In **P2**, we propose five critical factors that influence the distribution of security responsibilities between central teams or dedicated security experts and agile development teams. These factors aim to manage the balance between autonomy and control within LSAD settings. The insights are drawn from data gathered through a SLR and an interview study, as detailed in **P2**.

The first factor is the risk profile of the product being developed, which plays a pivotal role. Higher-risk products usually demand more stringent compliance efforts and extensive testing and verification before they can be deployed. This requirement directly affects the level of autonomy a team has, with more critical products necessitating tighter controls and potentially less autonomy for average agile teams.

Closely related to the product risk profile, the regulatory context and the company's risk appetite also significantly impact the autonomy granted to development teams. Industries with higher security demands enforce more rigorous controls, while a company's culture and size influence its overall risk tolerance. Larger organizations, in particular, tend to have more formal governance structures, limiting team flexibility. Startups, on the other hand, might be willing to take higher risks to ensure speed and improve market traction.

A team's security maturity is another critical determinant of autonomy. Teams with higher maturity may have a higher ability to handle releases and deployments independently, thus reducing bottlenecks and enhancing efficiency. This maturity is usually not static; it evolves throughout the project, necessitating periodic adjustments to security controls based on assessments of past security performance and self-evaluations.

Furthermore, the use of standardized infrastructure and development pipelines facilitates a reduction in manual security reviews. By enabling automated security testing and establishing quality gates, these tools allow teams to maintain high-security standards with less manual intervention, promoting greater autonomy. Furthermore, automation makes it possible for even less experienced developers to contribute effectively to application security.

Finally, constraints such as budget and time schedules also play a crucial role, influencing the

configuration of roles and the selection of security activities. These constraints can limit the extent of autonomy that teams can exercise, underscoring the need for efficient security integration into agile processes.

We provide more results on how these factors interact with each other and how they can be combined in one comprehensive approach to achieve a suitable autonomy control balance by answering our RQ3 in the next Section.

Summary. In our answer to RQ2, we present roles, activities and key influencing factors to balance the autonomy and control tension. These include recommended security-related roles such as SCs and SEs, as visualized in the LoD model. We also evaluate and recommend security activities such as threat modeling or pair penetration testing based on agile, large-scale agile, and regulatory criteria. The key influencing factors to balance the autonomy control tension include the product risk, the standardization level, and the team security maturity.

5.1.3. RQ3: Adaptive Process and Collaboration Model to Achieve Security and Agility

Our fourth publication, **P4**, focuses mainly on answering our third RQ.

Research question 3 (RQ3)

How can a parameterized adaptive process and collaboration model integrate security in large-scale agile development to achieve both agility and security?

In particular, in addition to the generic organizational structure presented in the previous RQ, **P4** contributes a team autonomy assessment model and an adaptive collaboration model.

In the team autonomy assessment model, a team autonomy score is calculated based on the most important influencing factors identified in our previous research.

These influencing factors and the resulting score are then used as parameters for the adaptive collaboration model which shifts the selection of security activities and the involved roles and their responsibilities based on these inputs.

We describe both the autonomy assessment model as well as the adaptive collaboration model in more detail in the following. As a basis, we first present the goals of these solution artifacts.

Goals of the autonomy assessment and adaptive collaboration model.

As described in **P4**, to bridge the identified research gap and tackle the prevalent challenges when integrating security in LSAD, we established six objectives for our solution artifacts:

1. We aim to clearly define roles and security activities in a way that is instructive but not overly restrictive. This addresses the lack of specific guidance for conducting security activities within LSAD environments, as noted by Moyón et al. [MMBK21] and van der Heijden et al. [vBS18], and aims to improve collaboration with non-development functions, as highlighted by Kalenda et al. [KHR18].

2. Our goal is to achieve security compliance while maintaining a balance between autonomy and control. This involves minimizing unnecessary controls to reduce overhead and avoid bottlenecks, addressing the need for a more flexible governance approach.
3. We seek to ensure that our approach integrates well within existing scaling agile frameworks, enhancing their practical applicability. This is crucial since current frameworks do not sufficiently address security concerns or the tension between control and autonomy, as identified in our previous work [NSM23].
4. Enhancing the security awareness and expertise of agile teams is essential, responding to the reported shortfall in these areas [MMBK21].
5. We advocate for a shift in responsibility towards agile teams, making security governance more compatible with agile methodologies. This shift aims to empower teams with greater autonomy while ensuring they meet compliance standards.
6. Finally, we intend to guide the focused allocation of security resources effectively, mitigating the impact of the widespread shortage of security professionals [MMBK21].

With these objectives in mind, we aimed for a structured yet flexible approach to embedding security practices within LSAD, ultimately enhancing the security posture while accommodating the dynamic nature of agile development.

Team Autonomy Assessment Model.

To navigate the balance between necessary control and granted autonomy within LSAD environments, we propose a model that assesses team autonomy based on three critical factors: protection need, team security maturity, and standardization level.

The protection need, influenced by the risk profile of the software product, regulatory demands, and the organization's risk appetite, dictates the required security measures and thus, the autonomy level. This assessment is a collaborative effort between the product owner and a SE, ensuring both business and security perspectives are considered. We categorize protection needs into four levels to ensure deliberate choice, guided by risk analysis and the potential impact of predefined damage scenarios, and prevent stakeholders from simply choosing a middle option.

Team security maturity, reflecting a team's ability to create secure and compliant software, suggests that higher maturity grants more autonomy. This maturity is evaluated through methods like self-assessments, external stakeholder evaluations, and (semi-)automated metrics, with periodic updates recommended to keep assessments up to date with minimal additional work. More details on such a maturity assessment are included in our answer to RQ4 in Section 5.1.4.

The level of standardization, identified as a significant factor through expert interviews, encompasses the use of standardized components, infrastructure, tooling, and security-related activities. This includes aspects such as reusable security components, pre-configured development environments, and standardized security practices, which collectively enhance a team's autonomy by providing a foundation of proven practices and tools.

To synthesize these factors into a practical team autonomy score, we assign each factor a proficiency level, which ranges from very negative to very positive impacts on autonomy. Summing

up these values, teams are classified into low, medium, or high autonomy categories, as shown in Figure 5.5.

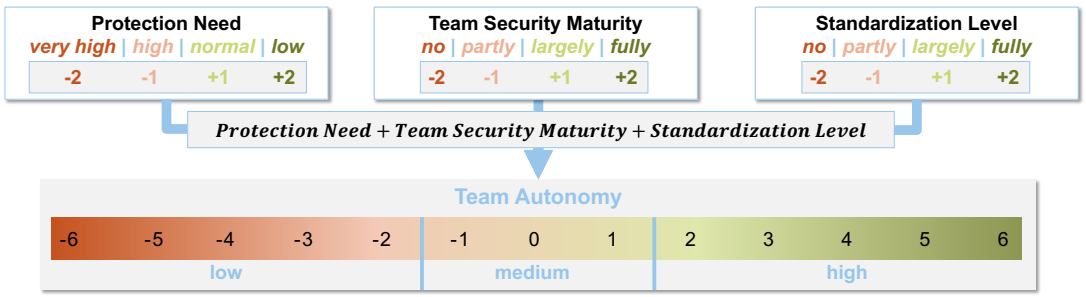

Figure 5.5.: Team autonomy assessment calculation including exemplary thresholds based on P4 [NSFM24]

This model allows for the compensation of high protection needs with high-security maturity and standardization, advocating for a dynamic balance that encourages continuous improvement in security practices and team capabilities. We deliberately chose the scoring to be as simple as possible since this was identified as one of the critical success factors for the applicability of our artifact in the expert interviews. Organizations are encouraged to adapt this scoring logic to fit their specific context and requirements, aiming to foster greater team autonomy, reduce process bottlenecks, and motivate ongoing security enhancements.

Adaptive Collaboration Model.

Our developed adaptive collaboration model is designed to efficiently incorporate security activities into development processes within our proposed organizational structure, dynamically adjusting roles and the frequency of these activities based on the level of team autonomy. The model, visualized in a generic version in Figure 5.6, assigns specific security tasks to roles through a color-coded system, indicating the necessity of activities based on the team's autonomy.

This allows for flexibility in the execution of security measures, where higher autonomy levels lead to a decrease in mandatory collaboration with SEs, less dependence on the SC, and fewer compulsory security activities.

To organize these activities, the model employs a categorization derived from established secure development life cycles, further refining the distinction between direct execution, accountability, and support for various tasks. It introduces the concept of "activity buckets," grouping similar security activities to streamline their distribution over time, thereby avoiding the need for their execution in every sprint or deployment. This approach enables teams to cover all necessary security practices systematically across different cycles.

The model outlines a range of security activities, including threat modeling, penetration testing, and code reviews, among others. It suggests customizing the execution frequency of these activities according to specific triggers or regulatory timelines, rather than strictly following sprint schedules. This flexible structure aims to guide teams in engaging in relevant security tasks appropriate for each stage of development, emphasizing the model's role as a catalog of activities rather than a linear process. The selection of particular activities and their detailed

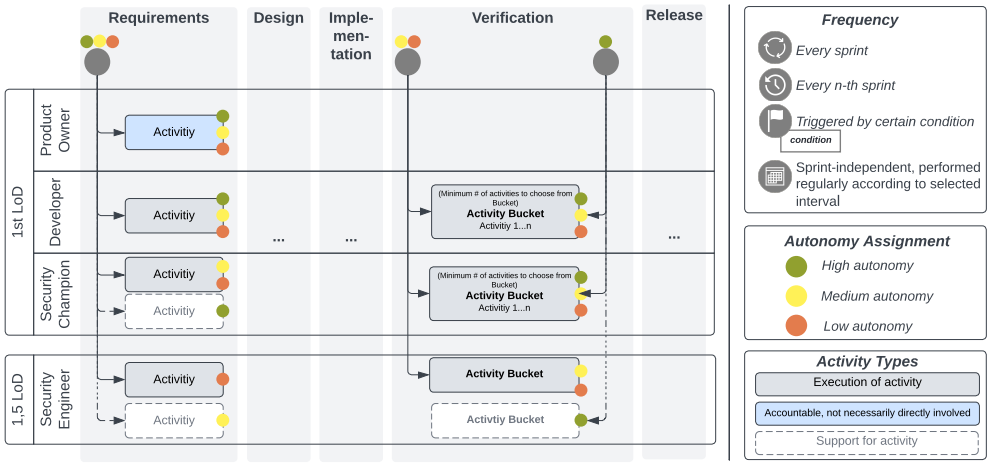

Figure 5.6.: Generic structure of the collaboration model based on P4 [NSFM24]

application is explored in detail in **P3**, offering in-depth insights into tailoring security practices to fit the unique needs of LSAD environments. The evaluation results of the models are found in **P4**.

Summary. In answering RQ3 we propose a comprehensive security governance and compliance in LSAD approach that leverages the previously identified influencing factors such as product risk and team security maturity to adapt the level of control by central security teams and supporting roles such as the SE to the requirements of the individual teams and their product context. This approach tackles the autonomy and control conflict and the identified challenges.

5.1.4. RQ4: A Maturity Model to Assess a Team's Capabilities to Develop Secure and Security-Compliant Applications

Our final RQ, RQ4, is mainly answered by our publication **P5**.

Research question 4 (RQ4)

How can a maturity model be designed and implemented to assess a team's capabilities to develop secure and security-compliant applications?

Since our solution approaches for improved security integration in LSAD rely heavily on the concept of security maturity of agile development teams, we focused on exactly that aspect in our publication **P5**. In **P5**, we contribute two main resulting artifacts, which we will explain the following: Ten team security maturity criteria that describe the capability of agile teams to develop secure and security-compliant applications in general, as well as a proposal for a specific model leveraging these criteria.

Team security maturity criteria.

In our study, we have developed a model to assess team security maturity across ten criteria. Inspired by the structure of Bishop and Rowland's [BR19] review on agility and security, we split these criteria into two categories: those not associated with the SDLC and those that are, with each category containing five criteria essential for a mature team. By not solely focusing on security competencies during development phases, we also encompass broader aspects that are important within the LSAD context, such as security awareness, team composition, and effective collaboration.

Figure 5.7 depicts an overview of the ten team security maturity criteria presented in **P5**.

	Criteria	Example assessment questions
Non-Associated SDLC	Awareness	Does the team understand the relevance of security (compliance) for their product?
	Composition	Does the team include at least one security expert, such as a Security Champion, for support?
	Knowledge	Is the team familiar with the security best practices and policies applicable to their product? Are there established mechanisms for knowledge sharing within the team?
	Training	Does the team (regularly) engage in improvements on security-related topics?
	Collaboration	Does the team frequently engage with security experts outside the team?
Associated SDLC	Activities	Does the team have procedures to identify security threats and vulnerabilities, both manually (e.g., code reviews) and through tools (e.g., SAST and DAST)?
	Development	Does the team promptly address identified vulnerabilities? Does it establish and uphold quality gates?
	Documentation	Does the team create consistently structured security documentation with minimal overhead?
	Product	What is the level of security quality and compliance of the products developed by the team?
	Responsibility	Does the team consider security as a requirement and assign corresponding responsibilities?

Figure 5.7.: Overview of the ten team security maturity criteria based on P5 [NWM24]

Regarding SDLC-associated criteria, we look at the team's engagement in security activities throughout the development process, their development practices aimed at preempting security issues, the quality and integration of security documentation, the security quality of the product as evaluated through audits or other assessments, and the clarity and distribution of security responsibilities within the team.

For criteria not associated with the SDLC, we examine the team's security awareness, its composition including roles like a SC, the security knowledge within the team, the extent of security training, and the team's collaboration with external stakeholders like SEs or information security officers. These aspects are crucial for a team's capacity to handle security challenges effectively and integrate security measures into their workflow.

Our approach suggests assessing these team security maturity criteria through a mix of self-evaluations, external evaluations, and the utilization of semi-automated metrics, advocating for regular updates to these assessments to ensure they remain reflective of the team's current capabilities. This structured approach allows for a nuanced evaluation of a team's maturity in handling security, aiming to foster an environment where security is seamlessly integrated into agile development processes. We apply these thoughts in the creation of our own model in the following.

Team security maturity model.

The TSMM offers a structured approach to assess and improve the security maturity of development teams in creating secure and compliant applications.

Designed with simplicity and practical application in mind, the TSMM employs a maturity grid that uses textual descriptions for each maturity level, facilitating easier self-assessment and classification by development teams. The model spans four levels across various topics, intentionally avoiding an odd number of options to eliminate a default "middle" choice and encourage a more thoughtful evaluation.

The TSMM structure includes pillars, domains, and topics, with three main data sources for determining maturity scores: self-assessments by teams, assessments by external roles, and the usage of semi-automated metrics.

We emphasize self-assessments as a key component, with teams encouraged to evaluate their security knowledge, execution of security activities, documentation practices, pre-release security measures, and collaboration and coordination concerning security. This decentralizes the assessment process, improving essential areas such as security awareness, informed activity selection, minimal documentation overhead, integration of quality gates, and defined security responsibilities within teams. External assessments complement self-evaluations by providing an outside perspective on the team's security maturity, focusing on audit practices and cultural aspects like continuous improvement and collaboration. SEs working with multiple agile teams are ideal for conducting these assessments, offering insights into the team's processes and culture.

Finally, the third pillar, automated assessments, leverages data from security tools, tailored to each organization's specific context. This includes analyzing security testing tools, manual test results, and auxiliary metrics such as security requirements fulfillment and engagement in security training, to objectively estimate a team's maturity level.

Each pillar contains multiple domains, and each domain is detailed by multiple topics. Table 5.6 shows an example excerpt of one of the TSMM domains, which revolves around knowledge building and sharing.

Table 5.6. Excerpt of the TSMM knowledge domain based on P5 [NWM24]

ID	Topic
K1	We understand the significance of security in the context of our product.
K2	We are aware of and adhere to internal and external standards relevant to our product.
K3	We facilitate knowledge-sharing sessions among team members.
K4	We identify and plan to rectify security knowledge gaps.
K5	We are aware of and use standardized components for secure development.

Overall, the TSMM aims to foster a comprehensive understanding of security maturity within

development teams, promoting a culture of continuous improvement and adaptation to evolving security challenges.

We implemented the TSMM in two formats: as an Excel spreadsheet and as a web application prototype. Excel's ubiquity and ease of use make it accessible for most business users, allowing straightforward customization without the need for programming. On the other hand, while the web application prototype entails much higher development and operating costs, it offers substantial benefits. It allows us to provide an API for integrating with other tools and the build pipeline as a whole, for example, to achieve an integration into existing quality gates. This allows for checks whether necessary assessments are completed or a certain maturity threshold is reached, for example, to deploy new changes to production. In addition, we are able to easily consume other APIs, e.g., for fetching metrics from security testing and analysis tools.

Another key benefit of the web application is the incorporation of gamification and community engagement elements, such as rankings, activity feeds, voting systems, achievements, and awards, which aim to foster a collaborative security culture.

These features, alongside statistics and other interactive tools, not only make the assessment process more engaging but also reduce manual input, thereby increasing the tool's acceptance and usage among teams. For a more in-depth exploration of the web application prototype, including screenshots and further elaboration on its functionalities, the outlook of this thesis provides more detailed information in Section 6.2.

Summary. For answering RQ4, we develop and propose a Team Security Maturity Model based on the ten identified Team Security Criteria which can be used to assess the capability of a team to develop secure and security-compliant applications. The core underlying idea is that the resulting team maturity score can be used in a way that more mature teams can then receive higher degrees of autonomy and responsibility, as described in answering the previous RQ.

5.2. Discussion of Key Findings

Our five publications contain a series of key findings (see Tables 5.7 and 5.8) that we describe in the following. First, we summarize and discuss all key findings for each publication and then conclude with an overarching discussion.

Publication 1: Key findings of our analysis on the state-of-the-art of security in LSAD.

We can summarize our comprehensive analysis of the current state of security in LSAD in eight key findings, described in the following.

The two first key findings revolve around two main challenges we identified in the area of security in LSAD through our SLR and expert interview study with nine different organizations. The first primary challenge is that there is a lack of qualified personnel with sufficient experience in both security (governance) and ASD (KF1.1). This challenge is amplified in LSAD due to the larger number of teams and the higher amount of experience and capacity required to achieve a suitable security in LSAD integration. The second challenge is the conflict between security

Table 5.7. Summary of key findings in publications P1-P3 [NWM22, NSM23, NKM23].

Pub.	Summary of Key Findings
P1	**(KF1.1)** There is a lack of qualified personnel with sufficient experience in both security (governance) and agile software development. **(KF1.2)** A key challenge is the conflict between security governance and team autonomy when coordinating many teams in a large-scale setting. **(KF1.3)** All analyzed cases have adopted dedicated security roles including central teams, team-internal, and team-external roles to enhance security measures. **(KF1.4)** Team-internal security roles like Security Champions are not utilized effectively in all studied cases, though they offer a strong potential for increasing team autonomy. **(KF1.5)** A robust DevSecOps pipeline with both static and dynamic security testing tools is essential to support agile teams in maintaining security with lower manual overhead. **(KF1.6)** Despite the predominance of top-down security governance, there is a growing shift towards bottom-up approaches, e.g., through security communities formed by agile teams. **(KF1.7)** Relying only on individual teams defining their own security standards can lead to unnecessary complexity and conflicts, underscoring the need for some top-down control. **(KF1.8)** Agile teams should influence security governance, and integrating self-governance through defined roles can balance autonomy with control, aligning with agile methodologies.
P2	**(KF2.1)** There is a growing demand for security integration in LSAD, though organizations face challenges stemming from the inherent tension between security and agile methodologies. **(KF2.2)** The identified key influencing factors can mitigate challenges by balancing control with the autonomy of agile teams at scale. **(KF2.3)** Security integration in LSAD faces unique challenges, such as unclear roles, responsibilities, and implementing security practices within iterative development at scale. **(KF2.4)** Most scaling agile frameworks do not adequately address the challenges of security integration, with DA and SAFe being exceptions that provide practical guidance. **(KF2.5)** Industry experts confirm the high relevance of integrating security within LSAD and emphasize the need for continued research in this area. **(KF2.6)** Further research is needed on the shift of security governance to agile teams and the balance of control between teams and central security units. **(KF2.7)** Investigating the benefits of LSAD for security might convince governance and audit roles of the advantages of autonomous teams, presenting a valuable research direction.
P3	**(KF3.1)** It is recommended to implement security roles before prioritizing the strengthening of security methodologies, as observed in the case company and related literature. **(KF3.2)** Methods like the security tagging system and peer security code reviews received positive evaluations from agile security experts. **(KF3.3)** In the case study, threat poker suited small-scale settings, security stories were ideal for LSAD, and the security tagging system fit regulated environments best. **(KF3.4)** The team-external security role proved most effective, while the Security Champion role showed promise but needed more development. **(KF3.5)** Persistent challenges in security knowledge exchange across teams improved slightly with the formation of a security community. **(KF3.6)** Our analysis presents applicable evaluation criteria for adopting security in LSAD that could benefit other organizations. **(KF3.7)** The identified factors that drive or hinder security adoption in LSAD aligns well with existing academic and practical insights.

Table 5.8. Summary of key findings in publications P4 [NWM24] and P5 [NWM24].

Pub.	Summary of Key Findings
P4	**(KF4.1)** Our adaptive approach leverages team autonomy to offer a resource-efficient way to manage security compliance within LSAD. **(KF4.2)** A significant shift towards autonomous and self-governing practices within large-scale enterprises is both feasible and beneficial for sustainable security integration. **(KF4.3)** Using team autonomy to tailor governance processes helps eliminate unnecessary controls, reducing bottlenecks and streamlining compliance to align with agile at scale. **(KF4.4)** Team autonomy is crucial for determining the roles involved in security activities and their frequency, allowing for a tailored approach that meets team-specific needs. **(KF4.5)** We recommend minimizing collaboration with team-external security stakeholders for teams with high autonomy to conserve resources and empower capable teams. **(KF4.6)** The proposed collaboration model was valued for its flexibility in distributing security tasks across roles and development phases, ensuring clear responsibilities. **(KF4.7)** Despite initial conflicts, with auditors seeking more control and agile practitioners finding the model too rigid, continued refinement led to an appreciated balance. **(KF4.8)** Governance and auditing roles particularly value the systematic and traceable methods our model uses to grant autonomy. **(KF4.9)** Agile practitioners appreciate the model for empowering them to enhance their skills and take on greater responsibility and autonomy. **(KF4.10)** Although our approach requires significant effort to implement, it has the potential to inspire incremental improvements to existing organizational procedures.
P5	**(KF5.1)** The ten team security maturity criteria help to develop or refine models for evaluating the security capabilities of agile teams. **(KF5.2)** Introducing maturity levels in LSAD alleviates the tension between autonomy and control, encouraging teams to enhance security competencies and granting more autonomy to mature teams. **(KF5.3)** The TSMM is designed to integrate with security governance as an enabler rather than a controller, aiming to empower development teams and improve their effectiveness. **(KF5.4)** Our approach combines self- and external assessments with semi-automated metrics to calculate team maturity scores, aiming for an unbiased assessment of team capabilities. **(KF5.5)** The TSMM increases organizational transparency and offers feedback that identifies security weaknesses and training needs within agile development teams. **(KF5.6)** The application and specifics of the TSMM must align with an organization's structure, risk profile, and technology usage. **(KF5.7)** The primary value of the TSMM lies in its general approach, framework and structure, rather than the specifics of the model. **(KF5.8)** The TSMM employs a grid-style approach for maturity evaluation, facilitating easier application and self-assessment through descriptive maturity levels. **(KF5.9)** The underlying principles and goals of the TSMM are adaptable to other product quality dimensions beyond security, such as architecture and UI/UX design.

governance and team autonomy when coordinating many teams in a large-scale setting (KF1.2). This is one of the core challenges that coined our remaining research and is the common theme that permeates all our five publications.

The third key finding is that we observed that all analyzed cases have implemented additional security roles (KF1.3), which aligns with existing literature recommendations. These roles vary in their details, encompassing central security teams, roles embedded within development teams, and external roles to bolster security measures.

In addition, despite the broad adoption of security roles, not all organizations are leveraging team-internal security roles such as SCs effectively. However, our analysis suggests that these roles are potentially the most effective long-term for enhancing autonomy and enabling teams to conduct more security-related activities independently (KF1.4).

With regards to security automation and testing, we found that a solid DevSecOps pipeline, equipped with both SAST and DAST tools, is indispensable (KF1.5). Such infrastructure is critical to support agile teams in maintaining rigorous security throughout the development process while reduing manual overhead.

In terms of governance, we found that security governance in the organizations we analyzed is predominantly driven top-down, contrary to the recommendations from recent literature and expert interviews that advocate for more bottom-up approaches (KF1.6). Nevertheless, there is a shift toward bottom-up initiatives, exemplified by development team members forming dedicated security communities, as observed in the organizations interviewed.

However, our study revealed that allowing individual teams to define their own security standards can lead to substantial and economically unjustifiable efforts (KF1.7). This decentralized approach can also cause conflicts of interest, suggesting that a certain level of top-down control is still necessary, particularly to comply with external audit requirements.

The eighth and last key finding of **P1** is the conclusion that, while top-down governance is prevalent, there is a notable need for agile teams to influence security governance decision-making. Integrating elements of self-governance through well-defined security roles can help balance autonomy and control, thus aligning security governance more closely with agile methodologies (KF1.8).

Finally, in **P1**, we also place our results in the context of related literature. For example, our research aligns our findings with the broader challenges of software security in LSAD as described by van der Heijden et al. [vBS18], particularly focusing on the integration of security within agile frameworks. We explore how the structure of agile programs, security governance, and security activities can address the challenge of aligning security objectives across distributed settings and clarifying roles and responsibilities. Additionally, our findings respond to the need for low-overhead security testing tools integration [vBS18], outlining practical solutions within agile environments. We also draw parallels between our results and established software security maturity models, such as the BSIMM [Syn]. Our analysis confirms similarities with BSIMM's identification of software security groups and extends the understanding by detailing not only team-internal roles but the more prevalent team-external roles in LSAD environments.

Publication 2: Key findings of our exploration on the current state of security governance and compliance.

In our analysis of security governance and compliance integration within LSAD environments, we have derived seven key findings, which we summarize in the following.

First, we observed a pronounced demand for security integration in LSAD environments, although organizations continue to encounter significant challenges. These challenges stem primarily from the tension between maintaining security and preserving agility, compounded by the complexities of scaled environments (KF2.1).

Second, the five influencing factors we identified might help mitigate these challenges by striking a balance between the control and autonomy of agile teams operating at scale (KF2.2). These factors include the risk profile, context and appetite, project budget and timeframe, standardization as well as team-related factors such as security capabilities.

Third, the integration of security practices with LSAD brings about unique difficulties, such as unclear roles and responsibilities, as well as issues in integrating and utilizing security testing tools effectively within iterative development processes (KF2.3). This complexity underscores the necessity for clearer strategies for integrating security effectively in LSAD settings.

Fourth, our review of established scaling agile frameworks reveals that most do not or do not adequately address these integration challenges, with the notable exceptions of DA and SAFe (KF2.4). These two frameworks at least provide useful initial ideas or guidelines that help address some of the core issues identified.

Fifth, feedback from industry experts confirms the practical importance of this research area and underscores the ongoing need for targeted academic research to address the prevalent challenges (KF2.5).

Sixth, a specific area highlighted for further investigation involves the shifting of security-related governance mechanisms to agile teams and determining the extent of control that should remain with central security governance units (KF2.6).

Lastly, exploring the advantages of LSAD for security, particularly in convincing governance and audit roles of the benefits of empowering autonomous teams, is seen as a promising area for future research (KF2.7). Such investigations could potentially validate and expand the applicability of LSAD principles to enhance the security posture of organizations.

Publication 3: Key findings of our security in LSAD case study.

Our case study delves into the adoption and integration of agile security approaches within a case company, revealing seven key findings.

Despite the greater emphasis on methodology approaches in academia, role and knowledge approaches remain crucial for successful security integration in LSAD, as evidenced by the case company's strategy of first implementing security roles to bolster security expertise, awareness and knowledge sharing before adopting specific security methodologies (KF3.1).

The evaluation of ten methodology approaches by agile security experts yielded favorable ratings

for methods such as the security tagging system, peer security code reviews, security stories with misuse cases, and threat poker (KF3.2).

In the context of the case study organization, threat poker showed potential for small-scale agile settings, security stories were most fitting for LSAD, and the security tagging system was deemed most suitable for regulated environments (KF3.3).

Regarding the adoption of agile security roles and knowledge approaches, the team-external security role emerged as the most successful and beneficial, according to our survey. The SC role also showed promise, although it requires further experience to fully realize its potential (KF3.4).

Challenges in cross-team and cross-tribe security knowledge exchange persist in the LSAD environment of the case company, although improvements were achieved through the establishment of a security community (KF3.5).

Our analysis not only highlights the beneficial and challenging aspects of adopting security approaches in LSAD but also presents evaluation criteria that could be applicable in other organizations and industries (KF3.6).

Moreover, we identified eleven factors that drive or hinder the adoption of security approaches in LSAD, with six aligning with existing literature, suggesting a significant overlap between academic and practical perspectives on agile security integration (KF3.7).

Publication 4: Key findings of our study on an adaptive approach.

In **P4**, we distill our research results into four key findings in the discussion section of the publication. Since those key findings are considerably more extensive than the individual key findings of our previous publications, we divide these four key findings into ten key findings in this thesis to achieve a more balanced scope of each finding compared to our other publications.

Firstly, our expert evaluations indicate that our adaptive approach, which assesses team autonomy to adapt governance processes, offers a resource-efficient basis for managing security compliance within LSAD (KF4.1). Despite the current preference for top-down governance in large-scale enterprises, our findings suggest that a significant shift toward more autonomous and self-governing practices is both feasible and beneficial, and facilitates sustainable security integration (KF4.2). Organizations can utilize team autonomy to tailor their governance processes, eliminating unnecessary controls to streamline security compliance and reduce bottlenecks, thus aligning more closely with agile methodologies at scale (KF4.3).

Team autonomy is critical in determining which roles participate in security activities and how often these activities occur, allowing for a tailored approach that adapts to the needs of each team (KF4.4). We recommend reducing collaboration with external stakeholders and security specialists for highly autonomous teams, thus conserving security resources and empowering teams to prioritize security more effectively (KF4.5).

The collaboration model we propose was deemed valuable by experts, providing a flexible distribution of security tasks across roles and development phases while ensuring clarity regarding responsibilities (KF4.6).

During our expert interviews, we found an interesting conflict. Auditors criticized a lack of structure and control, whereas agile practitioners viewed the model as overly rigid. However, through continuous refinement and discussion, both parties began to see the benefits of the approach, appreciating its ability to reconcile the need for structure with the flexibility and autonomy desired by agile practices (KF4.7).

Governance and auditing roles particularly appreciated the systematic and traceable methods the model employs to grant autonomy (KF4.8), while agile practitioners valued the model for empowering them to enhance their skills and take on greater responsibility and autonomy (KF4.9).

Finally, while the implementation of our approach is extensive and may require significant effort within organizations, it has the potential to inspire incremental improvements to existing procedures (KF4.10).

In **P4**, we also place our results in the context of related existing literature. For example, our research in **P4** tackles challenges described by Moyon et al. [MAR+20, MMBK21], such as the shortage of security practitioners, or the need for guidance on integrating security activities in LSAD, also described by Dännart et al. [DCB19] and Edison et al. [EWC21].

Publication 5: Key findings of our research on team security maturity.

Similarly to the previous subsection, for the sake of a similar key finding scope, we further divide the original five key findings from **P5** into nine total key findings, presented in the following.

The ten team security maturity criteria presented in our publication **P5** help to build new models or enhance existing ones for assessing the security capabilities of agile teams. These criteria have been validated by experts and acknowledged as comprehensively describing the maturity and capability of teams to develop secure and security-compliant software (KF5.1). However, we acknowledge that there might be room for uncovering additional criteria.

By introducing maturity levels, we alleviate the tension between autonomy and control within LSAD, encouraging teams to increase their security competencies and allowing more mature teams greater autonomy (KF5.2).

Therefore, our TSMM is designed to integrate with existing security governance structures more as an enabler than a controller, promoting development team empowerment, which should enhance overall security effectiveness (KF5.3).

Our findings show a preference for using a combination of self-assessments, external evaluations, and semi-automated metrics to calculate team maturity scores, offering a balanced view of team capabilities and reducing bias (KF5.4).

Overall, the TSMM increases organizational transparency and provides feedback, helping identify weaknesses and training needs in the security capabilities of agile development teams (KF5.5).

However, the application and detail of the TSMM need to align with an organization's specific structure, risk profile, and technology use (KF5.6). Hence, the value of the TSMM lies more in its general framework and structural approach than in the specifics of the model (KF5.7).

Our model fills a crucial gap by providing a grid-style approach for evaluating maturity instead

of a simple checklist, facilitating easier application, learning progression, and self-assessments through descriptive maturity levels (KF5.8).

The concept and underlying goals and ideas of the TSMM are also adaptable to various other quality dimensions of product development next to security, such as architecture and UI/UX design (KF5.9).

Our primary goal in presenting the TSMM is to encourage organizations to leverage team security maturity, allowing them to refine their governance processes and empower agile teams while maintaining robust security standards. In **P5**, we also contextualize our findings within the broader academic discourse, noting that enhancing team maturity not only addresses but also mitigates some of the unique security challenges associated with LSAD, such as aligning security objectives across distributed settings [vBS18].

Discussion of our research in an overarching context.

While the previous subsections summarized and discussed the key findings from each of our individual publications, this subsection presents an integrated discussion of our research as a whole.

The first crucial aspect to highlight is that the conflict between security governance and team autonomy in managing multiple teams in a large-scale setting (KF1.2), identified as one of the primary challenges in our initial publication **P1**, emerged as a central theme that significantly influenced our research across all five publications.

The prevalence and intensity of this problem stem from the cultural differences between agile development and security teams. As explained in the introduction, according to Wright [Wri14], based on the theories of Hall [Hal73], agile developers, as part of so-called low-context cultures, value trust and autonomy, whereas security teams, as part of high-context cultures, value control. This discord not only leads to increased costs and delays but, in our experience, also causes considerable frustration within the workforce. As researchers, we often directly felt the duality of the situation by interviewees of each side showing signs of anger and irritation when reporting on typical problems in their activities at the intersection of security and agile development at scale.

Agile team members often feel constrained and slowed by rigorous security governance processes, leading to a diminished sense of autonomy and, ultimately, reduced motivation. This frustration may even drive them to bypass security measures.

On the other hand, security teams experience their own set of challenges. They often face criticism and pushback for imposing security requirements and auditing their implementation, particularly when these interventions come too late in the development cycle to be efficiently integrated, thus causing further delays and increasing costs. In our observations, these late interventions were often caused by agile teams approaching security teams only briefly before aiming to release a new software iteration, which in many cases is too late. Additionally, the lack of necessary documentation and transparency in development and deployment processes can complicate compliance and audit tasks for security teams. In our experience, these discrepancies can even cause security teams to view agile development skeptically in general, perceiving it as overly focused on business requirements and value creation at the expense of security. This

might even result in security teams tending towards being more rigorous than necessary to counteract these trends, for example ending up with requiring overcomplicated documentation or inadequate security measures.

By sharing and discussing these observations, we do not aim to blame any of those groups. From the individual perspectives, their behavior is understandable. Though understandable, these observations highlight the fundamental issue that must be addressed at its root, which is why we chose the conflict between agile development at scale and security governance and compliance as the focus of our research.

As a second discussion item, we would like to mention that we also considered whether a structured, standardized, and complex solution approach, as detailed primarily in **P4** and **P5**, is appropriate for addressing issues deeply rooted in cultural differences in work styles, particularly within an agile development environment that prioritizes low-context cultures. This consideration aligns with what Weill from the MIT Sloan Center for Information Systems Research described as the "agility paradox" in 2006 [Wei06]. The term summarizes the study findings that increased standardization can actually enhance organizational agility and, even more, that it is a prerequisite for higher organizational agility. From our perspective, this paradox fits well with our research objectives; we aim to increase agility by granting more responsibility and autonomy to expert teams within a structured framework that still meets the requirements for security compliance, which necessitates a structured and transparent process. Additionally, our approach incorporates agile values, focusing on people and their interactions through a strong emphasis on roles and collaboration. Ultimately, our goal is to foster greater freedom and less control for teams, but within a clearly defined framework.

Interestingly, this also aligns with agile team members' feedback to our research, who expressed a preference for strong support and clear guardrails in security, leading us to our third and final overarching discussion point. Agile team members, often lacking experience in security, appreciated clear guidelines that save time and allow them to focus on their primary skills. Security tasks can be daunting for many developers, and they are often relieved to receive substantial support or delegate some of this responsibility.

This insight suggests that the perceived conflict between team autonomy and control may be partly "artificially" constructed. In our observations, it often emerges from certain stakeholders (e.g., agilists/methodologists), promoting team autonomy more vigorously than teams themselves desire, especially in areas like security. Based on our interactions, we are convinced that relying solely on team autonomy, self-organization, and self-governance for security may not be practical. While this approach might succeed in organizations with high maturity levels in both security and agile practices, it generally imposes excessive burdens on most organizations. These burdens include duplicated efforts and increased costs as each team individually addresses similar security challenges, leading to a diverse array of architectures, technologies, and security measures to manage.

Therefore, we recommend a balanced approach that integrates agile teams, team-external decentralized security experts, and centralized governance and compliance teams. The focus of governance efforts should be on "lean governance," which minimizes control and empowers teams to

self-manage. This approach helps avoid potential pitfalls of more traditional security governance, such as severe delays, escalated costs, and reduced employee satisfaction and motivation.

In the next section, we further discuss overarching aspects, but from the perspective of the implications for research and practice.

5.3. Implications for Research and Practice

Our extensive findings have a wide range of potential implications for research and practice. We would like to highlight four of these in more detail below.

Actionable insights from our broad state-of-the-art analysis.

First of all, our research results answering RQ1 provide detailed insights into how companies with LSAD environments integrate security. These results reveal challenges, best practices and solution approaches, as well as drivers and obstacles when adopting these approaches. We also shed light on the tendency that security governance is still mainly top-down driven, while our recommendation is to increasingly shift to bottom-up approaches, without forgoing central control completely.

Our findings also contribute to raising awareness of the key areas and challenges to consider when aiming to develop software securely at scale. For a successful integration of security in LSAD, it is essential to tackle four important areas, which are the structure of the agile program, security governance, integration of security activities, and tool-support and automation.

Practitioners could leverage these results by discussing, adopting, and tailoring the identified best practices in their organizations. Thereby, these insights can contribute to identifying the most pressing challenges for the respective organization, thereby allowing them to pinpoint their largest improvement potentials and specifically address these with the most relevant presented best practices. Overall, this enables practitioners and their organizations to enhance their security practices and strengthen their security posture.

Furthermore, the breadth of our state-of-the-art analysis through **P1**, **P2** and **P3** provides a rich resource for researchers interested in the dynamics of security within LSAD. It lays out a categorization of crucial security aspects in LSAD and serves as a foundation for further exploration into these critical areas.

Practical guidelines on agile security activities and roles.

Secondly, our research outcomes, particularly from RQ2, provide practical guidelines identifying the most effective roles and security activities for LSAD environments. As our research shows, SCs and SEs are valuable to improve the security posture of agile teams and improve collaboration and coordination of multiple teams and central security departments in LSAD environments. Security activities such as threat modeling suit the iterative development cycles of agile teams best. These actionable insights are crucial for companies either looking to adopt some of the agile security practices for the first time or aiming to refine their existing strategies.

The detailed evaluation in **P3** underscores the utility of specific security measures, outlining their benefits and limitations. This information may serve as a valuable resource for organizations in tailoring their security approaches to better suit their needs and improving their agile security decision-making process.

Moreover, the findings offer a solid foundation for future research, inviting researchers to delve deeper into the most promising security practices identified in our study. Our emphasis on generalizability, including the broad evaluation criteria that we developed and used, extends the applicability of our results beyond the scope of our initial investigation. This allows both practitioners and researchers to assess and possibly integrate new agile security approaches, enhancing the adaptability and effectiveness of security measures in agile development settings at scale.

Innovative models to balance the control and autonomy tension in LSAD security governance and compliance.

Thirdly, in addressing our third and fourth RQ, we contribute innovative models designed to navigate the tension between maintaining control and fostering autonomy within security governance and compliance in LSAD. Our adaptive collaboration model and TSMM may serve not only as a blueprint for initiating agile security governance approaches but also as a guide for refining and enhancing existing practices in organizations with new elements and insights drawn from our research.

We acknowledge that our collaborative approach is extensive, and its implementation within an organization may require significant effort. However, our approach could also inspire incremental changes to existing established procedures within organizations. In the spirit of iterative agile development, we also encourage incremental adjustments and continuous learning and adapting when it comes to security governance and compliance procedures.

With regard to the TSMM, which we present in **P5**, we consider the main contribution to be less in the specific details of the model and more in its overarching idea and structure. The TSMM is primarily intended to encourage organizations to realize the benefits of cultivating team security maturity. By doing so, they can refine their governance strategies to empower and support agile teams effectively, thereby boosting security posture without hindering the agility of the development process. Or seen the other way around, to adjust their agile development approaches at scale to better include security without compromising too much on agility. The perspective always depends on the maturity of the organization in both areas and their integration, but from our experience, an adjustment of both sides is necessary in most cases for a proper and balanced integration.

Bridging the gap between security and agility at scale.

Finally, to elaborate further on the aspect of briding the gap between security and agility at scale, we would like to mention an observation from our expert interviews and industry discussions as the fourth and last implication: Our research results are particularly interesting for two types of companies.

First are the companies in the midst of agile transformation, including those hesitant to adopt agile methodologies, often due to regulatory constraints or the sensitive high-risk nature of their business or industry. The second group consists of organizations with a high level of agile development maturity and experience, now seeking to bolster their security practices.

Our research introduces influencing factors and a collaboration model designed to serve both groups effectively. For the former, our model offers a way to make security processes more adaptable and integrated with agile practices. For the latter, it provides a systematic approach to integrate security, ensuring traceability and compliance without sacrificing the agility of development processes.

This duality is also obvious in the different roles that we studied and interviewed. While initially, some agile practitioners found our approach already far too rigid and process-based, auditors and security governance and compliance experts criticized the same approach for not possessing enough rigidity, structure and control.

These differing perspectives highlighted exactly the conflict we aim to address with our research. Through continuous dialogue and refinement, informed by the expert feedback, we observed a growing consensus on the merits of our proposed models. Governance and auditing professionals for example praised the systematic and traceable framework for enabling autonomy, while agile practitioners recognized the value in how the approach empowers them to enhance their expertise and receive increased autonomy based on a proven track record of maturity, thereby rewarding expertise with greater accountability.

As described in more detail in the discussion of our research in an overarching context in the previous section, we recommend a balanced approach that integrates agile teams, team-external decentralized security experts, and centralized governance and compliance teams, shifting the focus towards "lean governance". This approach helps avoid potential pitfalls of more traditional security governance, but also ensures avoiding the downsides of an approach driven purely by autonomous teams, which often results in excessive additional efforts.

In essence, our aim with our research is to bridge the gap between agile development and security, offering a pathway for organizations to harmonize these often still conflicting areas effectively, thereby addressing the prevalent challenges in securing agile software development at scale.

5.4. Limitations

In the following, we summarize the limitations of our research presented in publications **P1** to **P5** and the countermeasures we applied to address these limitations. In addition, we also describe some overarching limitations affecting our overall research.

Although our research in **P1** primarily involved an interview study, we structured our limitations in line with common limitations of case studies, as described by Runeson and Höst [RH09], which also apply to our work. To address construct validity, we clarified ambiguities directly during the interviews. To enhance external validity and improve generalizability, our interviews were grounded in scientific literature and conducted across nine organizations in five industries.

However, interviewing only one expert per company may not fully capture each organization's diversity, so we designed the interview guide questions with the aim of revealing general patterns and emphasized the focus on generalizability with the interviewees. Despite a relatively small number of interviews, we reached data saturation with the identified best practices, where additional interviews provided no completely new concepts. For reliability, all interviews were recorded, transcribed, and coded, with processes documented and validated by two researchers, following the interview guidelines by Kvale [Kva07].

In **P2**, we differentiate between limitations and countermeasures for our two primary research methods applied in the study, a SLR and an interview study. Regarding our SLR, we addressed the validity threats commonly relevant for SLRs in the software engineering domain highlighted by Zhou et al. [ZJZ+16], such as inappropriate search terms, databases, and selection criteria. We aimed to secure construct and internal validity by meticulously selecting databases, search terms, and publications. To counter publication bias [KC07], we included gray literature [FYLW20]. The validity of our interview study might be constrained by the selection of interviewees and possible researcher bias during data collection and analysis [MHS13]. We minimized these risks by using a semi-structured interview format, cyclic coding, and transcribing interviews to limit researcher bias. To improve external validity, we ensured a diverse expert sample and linked interview findings with literature insights, though more interviews could enhance generalizability.

In **P3**, we detail the limitations of using a case study as our primary research method, focusing on reliability, construct validity, and external validity [Kva07, Yin03]. To ensure reliability, we carefully documented the study's purpose, questions, theoretical framework, data collection methods, and results, following the case study protocol by Yin [Yin03]. We mitigated potential researcher bias in interview transcriptions and data interpretation by adhering to established guidelines for interview analysis. To enhance construct validity, we directly addressed ambiguities during interviews and confirmed key statements to prevent misunderstandings. We also used iterative feedback loops with pilot groups to refine the survey questionnaire for clarity.

Given the single-case study design, external validity was a concern. To address this, we developed a generic set of evaluation criteria applicable to various environments, including agile, LSAD, and regulated settings. The experience and insights from ten experts with backgrounds beyond the case company helped generalize our findings. These steps, along with parallels drawn from academic literature, suggest that our study's insights could be valuable to other organizations and researchers. However, due to time constraints and limited availability of industry experts, only some of the 27 identified approaches were assessed, which we pre-selected based on specific criteria described in **P3**.

In **P4**, to enhance the robustness of our results, we integrated expert interviews into both the development and final evaluation phases. To ensure the validity of our findings, we selected experts known for their extensive experience and diverse professional backgrounds, addressing concerns about expert representativeness [MHS13]. Recognizing the potential for researcher bias in data collection and analysis [MHS13], we adhered strictly to our semi-structured interview format and employed cyclic coding to analyze the data, with transcriptions helping to minimize any inherent researcher bias. A significant limitation of this study is the absence of experiments

and practical implementations of our approach, which is part of the outlook described in Chapter 6 of this thesis.

In **P5**, for our SLR and interview study, we explicitly adhered to the quality criteria and addressed the typical limitations presented by ACM SIGSOFT [SIG24] as part of the empirical standards for software engineering research. Although interviews were instrumental in evaluating our results, the restrictive time frame limited an exhaustive exploration of each topic and the practical application of the model within an LSAD environment. This inability to observe the direct impact on development teams' security maturity is a notable limitation, which we plan to address in future research. To enhance reliability, all interviews were recorded and transcribed, and we employed systematic content analysis and classification as per Kuckartz [Kuc14]. Construct validity was supported through the use of semi-structured questionnaires, pre-tested for comprehensibility, with ambiguities clarified during the interviews [RH09]. Nonetheless, the semi-structured nature of the interviews could lead to potential deviations and imprecisions. Finally, to ensure the validity of the designed artifact from our DSR process, we applied guidelines from Lukyanenko et al. [LEP14].

As a more general limitation affecting all of our studies, we would like to highlight the challenge of recruiting organizations that are willing to participate in studies about a sensitive topic such as the security of software products, especially when the results are intended to be published to a broad audience or even the general public. Hence, many potential and highly suitable interviewees had to withdraw as they could not secure necessary approvals, despite our efforts to anonymize all data and present our findings in an aggregated, anonymized manner. Additionally, while we uncovered notable insights across various organizations, we were unable to include certain findings due to their sensitive nature, which could potentially reveal sensitive information, such as indications about security postures, vulnerabilities, or reputational damages. Nonetheless, we have incorporated the implications of these insights into our solution artifacts to address the typical challenges of security in LSAD.

CHAPTER 6

Future Work

Our research produces various avenues for future work, which we summarize in Table 6.1 and briefly describe in Section 6.1. In this section, we also explain which future work items we already addressed ourselves in ongoing research and how our five publications are connected from the perspective of future work. In addition, we present larger avenues of ongoing and possible future work related to the research presented in this thesis in the subsequent sections. Section 6.2 outlines our ongoing web application tool support research, while the remaining sections dive deeper into possible future work regarding security metrics (Section 6.3), self-assessment guidelines (Section 6.4), a security compliance metaframework for software development (Section 6.5), as well as exploring the benefits of agile development concerning security (Section 6.6). These ongoing and future work opportunities mainly revolve around expanding our contributions III and IV.

6.1. Summary of Future Work Based on P1-P5

In the following, we will summarize future work opportunities based on our respective publications **P1** to **P5**.

In **P1**, we propose that future studies could further analyze and even quantitatively assess the impact and effectiveness of the best practices we identified in the four categories of organizational structure, security governance, security activities, and automation. We tackle part of this proposed future work ourselves. Specifically, with **P2**, we dive deeper into security governance, whereas with **P3**, we evaluate security activities in more detail. Given the complex nature of LSAD security, additional research could uncover and delve into additional aspects beyond these four categories we have outlined in **P1**. We also propose a shift towards more bottom-up security governance, suggesting a deeper analysis or evaluation of current governance models.

We also suggest that further studies could examine how mature software development models influence the adaptation of security governance and compliance processes in scaled agile environments. This further research might reveal that more mature development teams are better equipped to manage their security independently, potentially reducing the need for stringent top-down control. We addressed this potential further research stream with our publications **P4** and **P5**.

In **P2**, we suggest that future research could explore the factors influencing team autonomy that we have identified in more detail, possibly also discovering additional crucial factors, and develop methods for their assessment. This research could also examine how security governance and compliance processes can be specifically tailored based on the level of team autonomy. Furthermore, future studies could assess the impact of these adaptations within LSAD environments to evaluate their effectiveness and efficiency. Again, we address these future research possibilities with our own research, especially **P4** and **P5** address the question of how security governance and compliance can be adjusted based on the level of team autonomy.

In **P3**, we note that to enhance the generalizability of the results and augment empirical evidence, future research could explore the adoption of agile security approaches across different companies and industries. Furthermore, there is a particular need for studies focused on security activities in regulated LSAD environments, where existing research is limited. Additionally, examining both the driving and hindering factors in the adoption process and refining evaluation criteria for agile security approaches would benefit researchers and practitioners alike. Conducting experiments or a long-term field study to validate the effectiveness and usefulness of the top-rated security approaches also presents a compelling research opportunity.

In **P4**, future experiments should assess the practicality of our approach, initially focusing on a small number of teams to establish feasibility before extending to larger settings. Longitudinal studies could investigate whether more mature teams can effectively self-govern their security posture with reduced control. Evaluation metrics might include the frequency and severity of vulnerabilities and incidents, as well as the average time taken to resolve security issues. Additionally, we deem is important to consider key metrics reflecting the success of agile development efforts. Future adaptations of the approach could also address other cross-cutting concerns and non-functional requirements beyond security.

In **P5**, since our evaluation relied solely on expert interviews, future research could apply and analyze the TSMM in actual applications in LSAD settings to assess its effectiveness in measuring team capability and its impact on the autonomy-control tension in practice. As self-assessments form a crucial component of the model, additional studies should explore the success factors and develop guidelines for self-assessments within the context of agile team security maturity. Additionally, the automated aspect of the model presents significant research opportunities, particularly in determining the most accurate and cost-effective methods for predicting team maturity and capability.

Table 6.1. Overview of future work items based on P1-P5 [NWM22, NSM23, NKM23, NWM24, NSFM24]

No.	Future Work
P1	• Study the relative impact and effectiveness of recurring best practices identified in our research. • Investigate further aspects of security in LSAD beyond the four categories already identified. • Conduct in-depth studies or evaluations of existing approaches to shift toward more bottom-up security governance. • Research the influence of secure software development maturity models on adapting security governance and compliance processes to agile at scale. • Examine whether more mature development teams can effectively self-govern their security posture and reduce the necessity for top-down control.
P2	• Investigate and possibly identify additional factors that influence team autonomy and develop methods for assessing these factors. • Analyze how security governance and compliance processes can be adapted based on team autonomy levels. • Assess the effectiveness and efficiency of these adaptations within LSAD environments.
P3	• Investigate the adoption of agile security approaches in different additional companies and industries to improve result generalization. • Prioritize research on security activities suitable for regulated LSAD environments due to the scarcity of existing studies. • Further explore the factors that drive and hinder the adoption process of agile security approaches. • Develop and refine evaluation criteria for agile security approaches to aid both researchers and practitioners. • Validate the acceptance and usefulness of highly rated security approaches through extended experiments or field studies.
P4	• Assess the practicality of the adaptable collaborative approach, initially focusing on a smaller number of teams before extending to larger settings. • Investigate in a longitudinal study whether the hypothesis holds true that more mature teams can actually manage their security posture with less oversight. • Employ metrics such as the frequency and severity of vulnerabilities, the mean time to resolve security issues, and key agile development success metrics when evaluating the approach. • Consider adapting the approach to include other cross-cutting concerns and non-functional requirements beyond security.
P5	• Future studies should apply the TSMM in LSAD settings to evaluate its effectiveness in real-world applications, especially its impact on balancing team autonomy and control. • Investigate the success factors and establish guidelines for conducting self-assessments in agile team security maturity contexts. • Further examine the accuracy and cost-effectiveness of automated methods for predicting team maturity and capability within the TSMM framework.

6.2. Web Application Tool Support

To further improve the applicability of our approach, we have been developing a web application tool support that simplifies and supports the application of our solution artifacts presented in **P4** and **P5**. Its main purpose is to reduce the effort for data collection for self-assessments and security metrics, increase participation by employing gamification features such as scoreboards and achievements, as well as acting as a central hub for guidelines in multiple product quality areas, such as security and IT architecture. It is also already deployed in case study environments and has been actively used and evaluated by hundreds of users from agile development teams, security experts, as well as other product quality areas such as UI/UX, architecture, and IT operations. In future work, this application could be deployed in additional case study environments, further evaluated, and the results analyzed and published. In the following, we provide a short overview of the application functionality, including screenshots to better visualize the application for readers.

Self-assessment features.

Since our solution approaches presented in **P4** and **P5** rely heavily on self-assessments, the core functionality of our web application tool support focuses on supporting these self-assessments. Figure 6.1 shows the application's dashboard.

The dashboard enables users to see all their applications and their respective self-assessment statuses and progress in four different product quality areas at a glance. Teams can score points with their applications. The more relevant guidelines they fulfill in the four areas, the higher

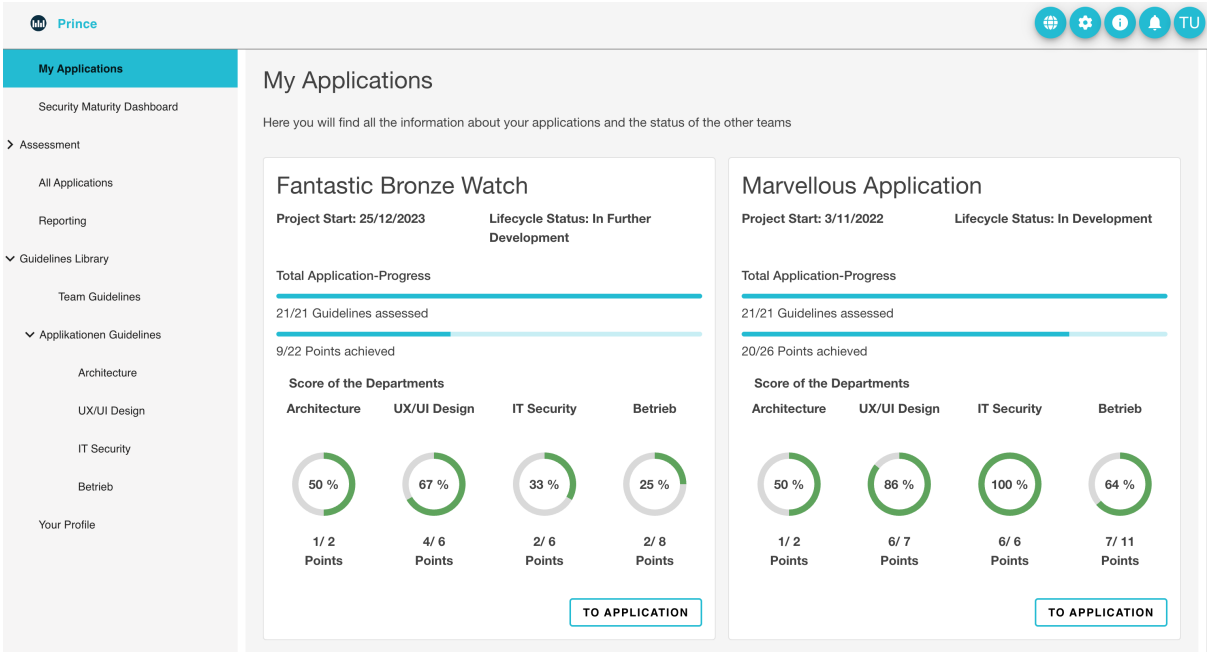

Figure 6.1.: Self-assessment dashboard of the web application

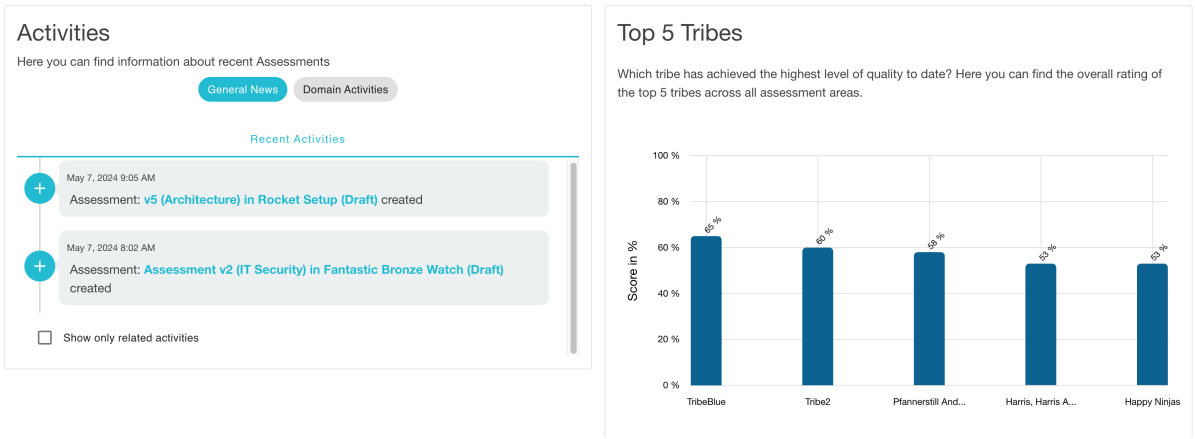

Figure 6.2.: Scrolling down on the self-assessment dashboard of the web application

their score will be. They can also justify not adhering to certain standards without any deduction if there are good reasons for deviating from the proposed standards.

Figure 6.2 shows the dashboard when scrolled further down, revealing additional information such as the top tribes, a first glimpse at one of our gamification features and community aspects aiming to motivate development teams, as well as an activity feed.

For each application, there is detailed helpful information available. For example, we show all important roles, such as product owner or SC, with their contact information, the current lifecycle status of the application, their respective tribe and team, and further details. The core functionality lies in supporting self-assessments for all of the applications of a team. Figure 6.3 shows the self-assessment view in the security category. All of the guideline questions, their answers, and associated scores are fully customizable.

The application further simplifies self-assessments by filtering only the self-assessment questions that are relevant to the current team and product. This is done by answering a short questionnaire about the characteristics of the application in development, and underlying matching of those criteria with all available guidelines from the guidelines library. Based on this matching, only the relevant guidelines get chosen and displayed for the self-assessments. The simplest example to explain this procedure is the question of whether the team in question develops a user frontend or a pure backend application or service. If there is no frontend, then the user interface and experience guidelines are filtered out. Further tags could, for example, revolve around the technology in use, the employed architecture, or other important aspects that allow filtering. These tags and filter questionnaires are fully customizable and can be adapted to the specifics of any organization employing the web application tool support.

Team security maturity features.

There is also functionality that revolves mainly around supporting the approach presented in **P5**, which is to measure the capability of agile teams to develop secure and security-compliant applications. We enable this by providing an additional dashboard specifically designed to access the most important features regarding team security maturity. Screenshot 6.4 shows the

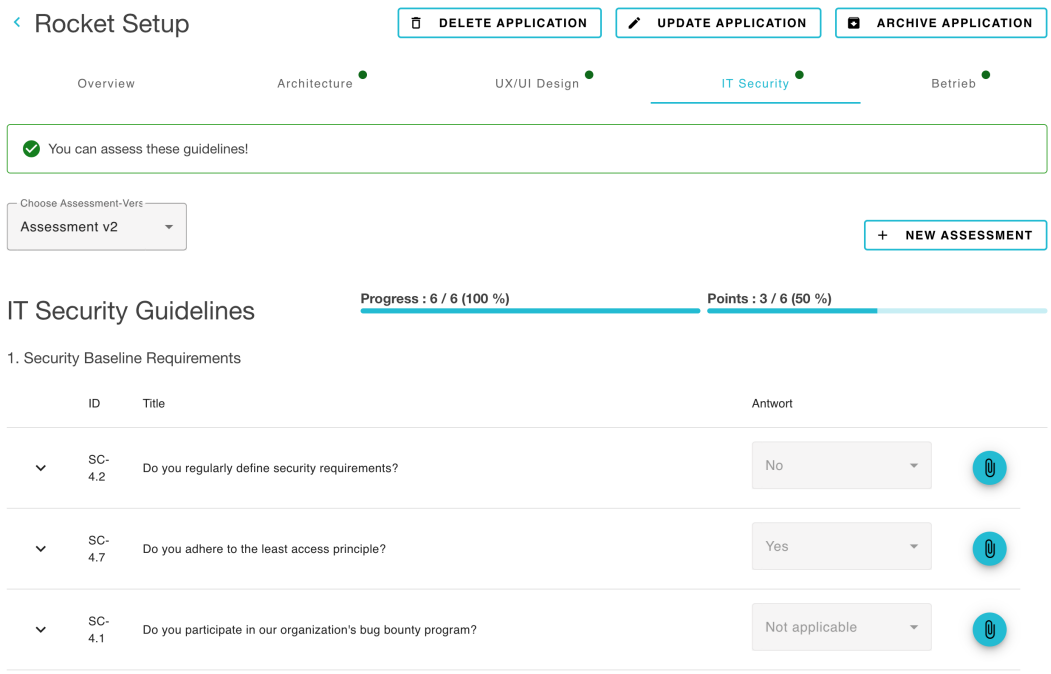

Figure 6.3.: Self-assessing an application in the security domain

dashboard and its functionality, mainly revolving around assessing and displaying the security maturity of development teams. It not only gives the application detailed insight into the resulting calculated maturity score for each team and its components, but also allows deeper analysis into the individual metrics of the three categories, which are internal and external assessment as well as automated metrics.

Guideline library and reporting features.

The application is also intended to serve as a centralized repository and "single-point-of-truth" for all ASD principles and guidelines within an organization. It allows users to create and manage guidelines across various categories, which can be organized and filtered using tags. The guideline library within the application organizes all available guidelines by key product quality areas, such as security or IT architecture. Users can easily search, filter, and access detailed information for each guideline. These guidelines can not only include thorough descriptions but may also provide links to additional resources such as code snippets or detailed implementations, typically stored in a company wiki or version control system. This setup enhances the practical application and relevance of the guidelines.

Moreover, the application features multiple reporting capabilities. It can generate various statistics, such as the total number of active users or the frequency of conducted self-assessments across different categories over time. Additionally, it offers specific reports for guidelines, displaying the number of teams and applications that comply with or fail to meet certain guidelines. This feature is particularly useful for identifying ineffective guidelines, which often possess widespread

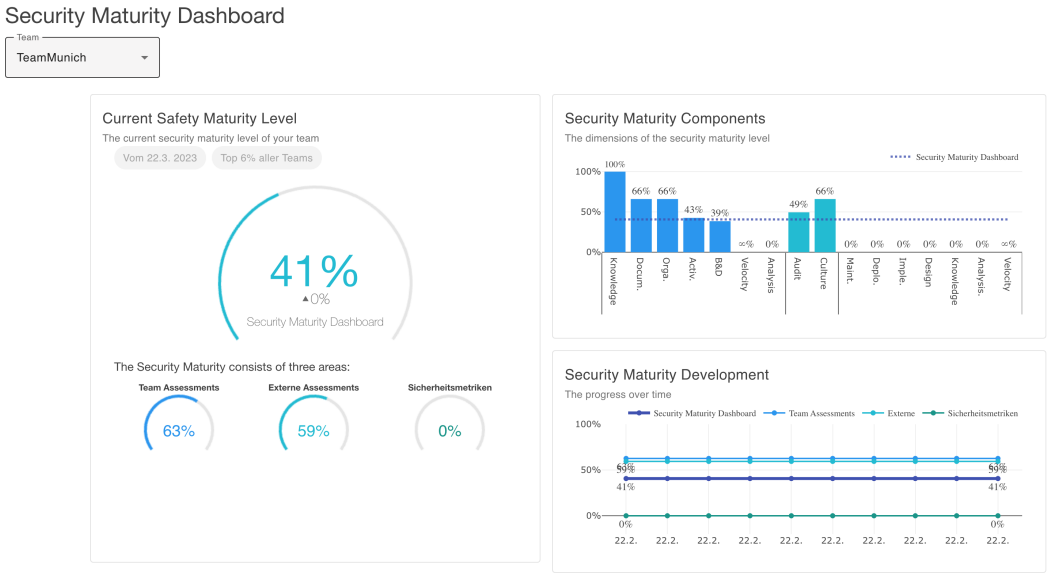

Figure 6.4.: The web application support of our team security maturity approach

non-compliance, meaning almost no teams fulfill the guideline, which may suggest issues such as lack of clarity or impracticality during implementation.

Gamification and collaboration to increase participation.

In our application, we have integrated gamification and collaboration features to enhance user engagement. Notably, we implemented a scoreboard that ranks teams and tribes based on their performance, along with rewards and badges that users can earn and display on their profiles. We decided to display only the top-ranking teams instead of also showing the lower parts of the scoreboard to prevent highlighting teams that currently perform the worst. An extensive activity feed provides updates on the latest events within the application, which users can filter to see activities specific to their teams and tribes or general updates. Additionally, a notification system highlights critical activities and news, prompting specific actions from users, such as reviewing new assessments for which they are responsible. These features are designed to increase interaction and foster a collaborative environment within the application.

Comprehensive admin functionality.

The web application offers an extensive admin menu that allows to manage and customize almost all important aspects of the application. For example, it allows to manage applications, users, teams, tribes, tags and tag groups, as well as customize all aspects of guidelines. It also provides usage statistics like number of active users and submitted assessments. It provides access to the security metrics and integration settings and allows to send bulk emails, for example when having to notify all SCs or product owners, or a specific filtered subset of such roles. Screenshot 6.5 shows an overview of the menu and the tag group management page as an example.

APIs for efficient integration into DevSecOps landscape.

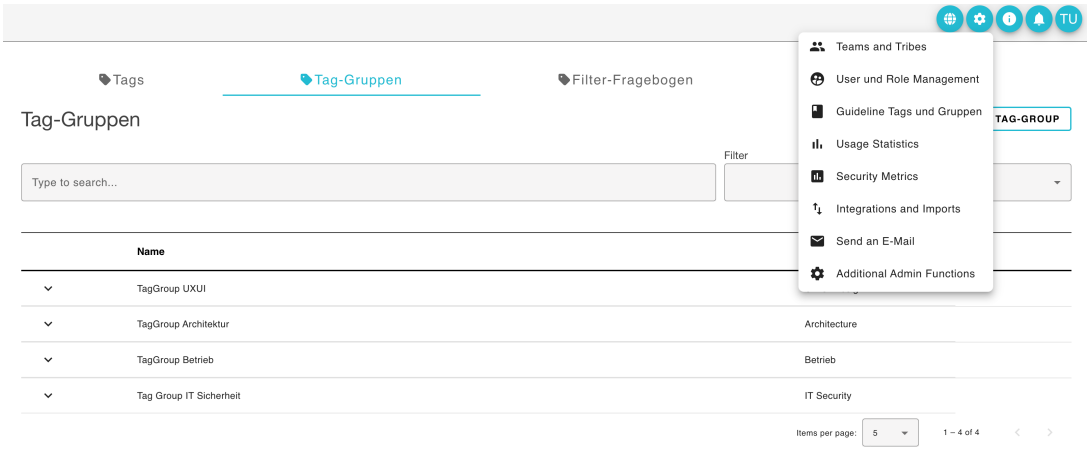

Figure 6.5.: The comprehensive admin menu of our web application tool support

The web application we developed both provides an API and consumes multiple APIs to enhance its functionality and applicability within existing development landscapes.

First, it allows for seamless integration into existing build and deployment pipelines through its API. For instance, during a build and deployment process, the API can be automatically accessed by the deployment script to retrieve self-assessment data for a specific application ID to check if assessments are complete and whether they meet the required quality thresholds. This integration enables the tool to serve as a quality gate, ensuring development teams are aware of and comply with relevant product quality guidelines tailored to their use cases and requirements.

Additionally, the application automates data synchronization by consuming APIs from systems like a configuration management database tool, such as ServiceNow. This automation imports and synchronizes essential application details and roles, such as Product Owners, reducing the manual effort and eliminating the need to manage application details in multiple places. The tool also integrates with Enterprise Architecture software such as ADOIT [Gro24] to automatically import crucial information, including development team structures and product risk levels.

Moreover, the application connects with product quality and testing tools like SonarQube [Son24] through their APIs. This connection pulls key metrics for each application, which are then displayed on the centralized dashboard of the web application and used to calculate team security maturity scores. By leveraging these metrics, the application reduces the reliance on self-assessments and mitigates potential biases, enhancing the overall effectiveness and accuracy of security assessments.

6.3. Security Metrics

As outlined in **P5**, team security capability can be gauged through security maturity scores that systematically evaluate both the team and its activities across various domains. Traditional as-

sessments typically depend on manual self-assessments or external audits using questionnaires, which can be biased and time-consuming. To enhance these assessments, we propose the integration of SSE metrics as an additional tool to measure team security maturity. These metrics quantify attributes such as the number of unmitigated vulnerabilities or the team's familiarity with security policies, providing continuous measurement throughout the SDLC.

Currently, there lacks a universally accepted set of high-quality metrics, and the existing ones are often unstructured. Moreover, contemporary security maturity models do not incorporate these metrics in calculating a security maturity score. To bridge this gap, we have begun developing a structured catalog of security metrics. For instance, metrics like the architecture component attack surface analysis rate, vulnerability scanning coverage rate, and mean time to resolve security issues are included. We follow a structured format similar to that proposed by Bouwers et al. [BvDV14]. Each metric in our catalog includes attributes such as a name, definition, expected value range, the rationale for its use, implications, and applicable contexts.

We aim to establish strict qualification criteria for these metrics and document them using a standardized catalog format. We are also exploring tools that can automatically measure these cataloged SSE metrics. Furthermore, we have begun incorporating these cataloged metrics into our maturity model from **P5** to complement traditional assessments and devise a method to derive a maturity score combining assessments and metrics.

Lastly, we have introduced a new module focused on security metrics into our web application support tool, as presented in the previous section. This module allows development teams and decision-makers to monitor security metrics and their maturity progression throughout the SDLC.

Further research could be invested into such a catalog, providing a useful basis for leveraging security metrics in the context of security in LSAD.

6.4. Self-Assessment Guidelines

In **P5**, we highlight the significance of self-assessments in evaluating team security maturity. Despite their utility, self-assessments come with inherent challenges and require careful consideration of several factors to ensure their effectiveness. Our ongoing research focuses on providing structured guidance for organizations implementing self-assessments within security contexts in LSAD settings. We suggest a series of critical aspects organizations must address to tailor self-assessments effectively. These include:

- Defining the objective, scope, and subjects of the self-assessment.
- Deciding whether assessments should be conducted independently by teams or with a moderator.
- Determining which team members should participate.
- Establishing the timing and frequency of assessments.

- Integrating the self-assessment process into the broader security governance framework of the organization.

Future research could further explore these aspects, delving into the nuances of self-assessment practices and their application in various LSAD environments to enhance their relevance and effectiveness.

6.5. Security Compliance Metaframework for Software Development

As security standards and regulations grow in importance, their role in securing applications and enterprises through comprehensive controls becomes crucial. Notably, a vast number of organizations are certified under the ISO/IEC 27001 information security standard [ISOb]. Such standards provide structured guidance on security controls and establish trust between developers and users of information systems [MC16]. Regulations like for example the requirements of the Federal Office for Information Security of Germany (BSI) for the banking and insurance sector in Germany legally bind relevant organizations to comply, with non-compliance leading to severe penalties, including potential loss of market license.

Moreover, adherence to standards such as the National Institute of Standards and Technology (NIST) Cyber Security Framework (CSF) or the Cloud Security Alliance (CSA) Cloud Controls Matrix (CCM) is increasingly required for business operations, not mandated by law but as a business prerequisite. For instance, cyber-insurance is more likely to be granted to organizations that meet recognized security standards [Eur16]. Certified compliance not only boosts credibility but also reduces the burden of security due diligence.

Given the significance of these standards, especially in the context of SSE within LSAD environments, future research should focus on a structured comparison of these standards and regulations. Current mappings of multiple security standards do not adequately cover software-centric standards and often lack in quality. Our future work aims to fill this gap by analyzing and comparing four categories of standards: information security standards, security regulations, secure software development standards including maturity models, and secure software development regulations. This analysis could reveal key insights into the implications of security compliance in LSAD settings.

6.6. Benefits of Agile Development for Security

Agile development methodologies have multiple advantages. For example, they offer extended and faster feedback loops and may lead to improved quality and better fulfillment of user needs [LSC20]. However, the focus of our research and our core publications so far has been mainly on the security challenges and problems within LSAD environments. Therefore, we have started to identify emerging patterns that highlight the potential benefits of LSAD for achieving secure and security-compliant applications.

These benefits, for example, include the utilization of backlog and pipeline audit trails for traceability of changes, faster response to new regulations, continuous iterative risk management, and enhanced security knowledge sharing through agile development communities.

Further investigating these advantages could bolster the reputation of agile methodologies among security experts and leverage the strengths of agile practices in enhancing security, rather than solely addressing its challenges.

CHAPTER 7

Conclusion

The growing adoption of agile methods at scale, originally not explicitly designed for scaled environments, presents unique challenges, particularly when combined with the increasing need for robust security measures due to rising incidents and their financial and reputational costs. This context underscores the necessity for effective security integration in LSAD settings. A central theme throughout our research, as highlighted across five core publications and this dissertation, is the tension between security governance and team autonomy faced by organizations managing a larger number of development teams in their LSAD environments. This conflict is a crucial challenge that has shaped our research direction and findings.

Our research addresses this challenge by exploring how to integrate security effectively within LSAD environments through our series of core publications, **P1** [NWM22], **P2** [NSM23], **P3** [NKM23], **P4** [NSFM24], and **P5** [NWM24], and achieves four contributions by answering four key RQs.

With our first contribution, we provide insights into the current state of challenges, drivers, recurring patterns, and opportunities for security integration. Our second contribution focuses on offering practical recommendations and best practices, including roles, responsibilities, security activities and their evaluation criteria, as well as factors that may be used to balance the tension between control and autonomy. While these two contributions are mainly achieved through publications **P1**, **P2**, and **P3**, our third contribution, mainly achieved through **P4**, introduces a comprehensive approach that synthesizes these elements into a collaborative and adaptable framework for effective security governance and compliance in LSAD settings. This approach leverages the roles, activities, and influencing factors identified in earlier research to enhance security integration.

Finally, resulting in our fourth contribution, publication **P5** delves into the concept of team security maturity, a key influencing factor identified in **P1** and **P2**, and applied in **P4** within our

overarching approach. Our research discussion argues that the degree of control and autonomy granted to agile teams should correspond to their capability to develop secure and security-compliant applications, advocating for adjustments based on team maturity levels. This focus on team capability aims to strike a balance between control and autonomy, tailoring security governance to the maturity and needs of each team.

Summary of Contribution I. Our first contribution provides a comprehensive analysis of security in LSAD environments, significantly enriching the current literature and industry understanding through a SLR and an interview study published in **P1**. We pinpoint four essential categories for security in LSAD: the structure of agile programs, security governance, adaptation of security activities, and tool support. These are detailed alongside 17 best practices based on industry input. In a subsequent study, **P2**, we refined our focus on challenges associated with security in LSAD, identifying 15 relevant challenges grouped into three categories: specific security challenges in LSAD, security challenges transferable from small-scale agile settings, and general LSAD challenges applicable to security contexts. We also reviewed existing agile frameworks' approaches to security, revealing a general scarcity of robust practices within LSAD-specific frameworks. To address these gaps, we explored enhanced LSAD approaches, agile methods, and SSE practices adapted for large-scale settings. A follow-up study, described in **P3**, analyzed the adoption of agile security approaches identified from academic literature through a case study in the finance sector. This extensive case study involved various research methods including interviews, document analysis, and surveys, leading to the identification of key drivers and obstacles in the adoption of agile security practices.

Summary of Contribution II. The second contribution enriches the first contribution with best practices that extend beyond observed patterns, providing deeper context, recommendations, and novel research insights. Notably, **P2** introduces five key factors that balance control and autonomy to mitigate security challenges in LSAD. Furthermore, **P3** evaluates 14 agile security methodologies through an extensive case study, assessing their effectiveness across various scenarios and contributing specific evaluation criteria. These criteria help assess the compatibility of security activities with agile methods, their suitability for large-scale environments, and their regulatory impact, aiding both organizational application and further research. In **P4** and **P5**, we extend our contributions to organizational structure and team success factors within LSAD. **P4** offers detailed organizational recommendations, outlining roles and responsibilities, while **P5** focuses on identifying factors that enable agile teams to achieve robust security practices. These publications provide foundational insights and practical recommendations for enhancing security integration in LSAD environments.

Summary of Contribution III. Our first two contributions already tackle some of the challenges identified, yet some issues such as the ambiguity around security roles, responsibilities, activities, and interactions in LSAD remain unresolved. These challenges necessitate a unified strategy to orchestrate various solutions into a cohesive approach, particularly to resolve the conflict between autonomy and control. **P4** introduces an innovative adaptive approach to security governance in LSAD, featuring a flexible organizational structure for security roles and an adaptive collaboration model that aligns specific activities with designated roles based on team autonomy levels. This model effectively balances autonomy and control while ensuring regulatory compliance. It builds upon the foundational challenges and solutions identified in

publications **P1** through **P3**, and uses team security maturity as one of the critical influencing factors to adapt the model. Team security maturity is researched in more detail in **P5** and provides the fourth and final contribution, summarized in the next paragraph.

Summary of Contribution IV. Building on our exploration of the autonomy-control conflict, **P5** focuses on team security maturity as a key factor in mitigating this conflict, exploring specific criteria that enable teams to develop secure and compliant applications. We have defined ten essential criteria for assessing team security capabilities, developed through a SLR and enhanced by expert interviews. We present these in a Team Security Maturity Model (TSMM), which demonstrates how to calculate a maturity score using a combination of self-evaluations, external assessments, and metrics from semi-automated sources like SAST and DAST tools. **P5**'s findings are valuable for the integration into the broader framework of **P4**, enhancing their relevance by considering additional factors such as product risk, thereby offering a comprehensive strategy to balance autonomy and control in large-scale agile settings.

In summary, our dissertation provides actionable insights and guidelines to tackle security integration in LSAD, addressing the challenges and research gaps that we identified throughout our research. With our adaptive collaboration and team security maturity model, we contribute innovative solution artifacts to balance the control and autonomy tension in LSAD and thereby bridge the gap between security and agility at scale. Further research may evaluate our approach through experiments in actual LSAD environments, and refine and study our web application tool support to improve the applicability of our solution artifacts.

Bibliography

[ABE+08] Julia Allen, Sean Barnum, Robert Ellison, Gary McGraw, and Nancy Mead. *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley Professional, first edition, April 2008.

[ADW17] Wasim Alsaqaf, Maya Daneva, and Roel Wieringa. Quality Requirements in Large-Scale Distributed Agile Projects – A Systematic Literature Review. In Paul Grünbacher and Anna Perini, editors, *Requirements Engineering: Foundation for Software Quality*, pages 219–234, Cham, 2017. Springer International Publishing.

[AI13] Ashley Aitken and Vishnu Ilango. A Comparative Analysis of Traditional Software Engineering and Agile Software Development. *2013 46th Hawaii International Conference on System Sciences*, pages 4751–4760, January 2013.

[AKS11] Scott W Ambler, Per Kroll, and IBM Software. Applying agile and lean principles to the governance of software and systems development. Technical report, IBM, 2011.

[AL19] Scott Ambler and Mark Lines. *Choose Your WoW!: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working*. Independently published, January 2019.

[All] Cloud Security Alliance. CSA Cloud Controls Matrix (CCM). https://cloudsecurityalliance.org/research/cloud-controls-matrix. (Online, last accessed: 25.04.2024).

[Amb08] Scott W. Ambler. Agile Software Development at Scale. In Bertrand Meyer, Jerzy R. Nawrocki, and Bartosz Walter, editors, *Balancing Agility and Formalism in Software Engineering*, pages 1–12, Berlin, Heidelberg, 2008. Springer.

[Amb09] Scott W. Ambler. Scaling agile software development through lean governance. In *2009 ICSE Workshop on Software Development Governance*, pages 1–2, May 2009.

[AN16] S. Hassan Adelyar and Alex Norta. Towards a Secure Agile Software Development Process. In *2016 10th International Conference on the Quality of Information and Communications Technology (QUATIC)*, pages 101–106, September 2016.

[AON] AON. Global Risk Management Survey 2023/2024. https://www.aon.com/en/insights/reports/global-risk-management-survey. (Online, last accessed: 24.04.2024).

[App24] AppSecure-NRW. AppSecure-nrw/security-belts. AppSecure.nrw, April 2024.

[ASM05] B. Arkin, S. Stender, and G. McGraw. Software penetration testing. *IEEE Security & Privacy*, 3(1):84–87, January 2005.

[Auta] German Federal Financial Supervisory Authority. Amended VAIT now available in English. https://www.bafin.de/ref/19621254. (Online, last accessed: 25.04.2024).

[Autb] German Federal Financial Supervisory Authority. BaFin amends its BAIT. https://www.bafin.de/ref/19620004. (Online, last accessed: 25.04.2024).

[Bar11] Steffen Bartsch. Practitioners' Perspectives on Security in Agile Development. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 479–484, August 2011.

[BAR13] Alan W. Brown, Scott Ambler, and Walker Royce. Agility at scale: Economic governance, measured improvement, and disciplined delivery. In *2013 35th International Conference on Software Engineering (ICSE)*, pages 873–881, May 2013.

[Bas16] Julian M. Bass. Artefacts and agile method tailoring in large-scale offshore software development programmes. *Information and Software Technology*, 75:1–16, July 2016.

[BBC+01] Kent Beck, Mike Beedle, Alistair Cockburn, Ward Cunningham, Martin Fowler, Robert C. Martin, Ken Schwaber, and Jeff Sutherland. Manifesto for Agile Software Development. https://agilemanifesto.org/, 2001. (Online, last accessed: 02.05.2024).

[BBCJ15] Dejan Baca, Martin Boldt, Bengt Carlsson, and Andreas Jacobsson. A Novel Security-Enhanced Agile Software Development Process Applied in an Industrial Setting. In *2015 10th International Conference on Availability, Reliability and Security*, pages 11–19, August 2015.

[BBSB17] Laura Bell, Michael Brunton-Spall, Rich Smith, and Jim Bird. *Agile Application Security: Enabling Security in a Continuous Delivery Pipeline*. O'Reilly Media, Sebastopol, CA, 2017.

[BC11] Dejan Baca and Bengt Carlsson. Agile development with security engineering activities. In *Proceedings of the 2011 International Conference on Software and Systems Process*, pages 149–158, Waikiki, Honolulu HI USA, May 2011. ACM.

[BGM87] Izak Benbasat, David K. Goldstein, and Melissa Mead. The Case Research Strategy in Studies of Information Systems. *MIS Quarterly*, 11(3):369, September 1987.

[BH06] John Brewer and Albert Hunter. *Foundations of Multimethod Research: Synthesizing Styles*. Sage Publications, Thousand Oaks, Calif., 2006.

[BH19] Julian M. Bass and Andy Haxby. Tailoring Product Ownership in Large-Scale Agile Projects: Managing Scale, Distance, and Governance. *IEEE Software*, 36(2):58–63, March 2019.

[Bha12] Anol Bhattacherjee. Social Science Research: Principles, Methods, and Practices. *Textbooks Collection*, January 2012.

[BJBC17] Martin Boldt, Andreas Jacobsson, Dejan Baca, and Bengt Carlsson. Introducing a Novel Security-Enhanced Agile Software Development Process:. *International Journal of Secure Software Engineering*, 8(2):26–52, April 2017.

[BK04] Konstantin Beznosov and Philippe Kruchten. Towards agile security assurance. In *Proceedings of the 2004 Workshop on New Security Paradigms*, NSPW '04, pages 47–54, New York, NY, USA, September 2004. Association for Computing Machinery.

[BLM09] Alexander Bogner, Beate Littig, and Wolfgang Menz. Introduction: Expert Interviews — An Introduction to a New Methodological Debate. In Alexander Bogner, Beate Littig, and Wolfgang Menz, editors, *Interviewing Experts*, pages 1–13. Palgrave Macmillan UK, London, 2009.

[BMSS18] Miklos Biro, Atif Mashkoor, Johannes Sametinger, and Remzi Seker. Software Safety and Security Risk Mitigation in Cyber-physical Systems. *IEEE Software*, 35(1):24–29, January 2018.

[BR19] David Bishop and P Rowland. Agile and Secure Software Development: An Unfinished Story. *Issues In Information Systems*, 2019.

[BS15] Dayanne Araujo Barbosa and Suzana Sampaio. Guide to the Support for the Enhancement of Security Measures in Agile Projects. In *2015 6th Brazilian Workshop on Agile Methods (WBMA)*, pages 25–31, October 2015.

[BSM20] Carlos Magnum M. Bezerra, Suzana C. B. Sampaio, and Marcelo L. M. Marinho. Secure Agile Software Development: Policies and Practices for Agile Teams. In Martin Shepperd, Fernando Brito e Abreu, Alberto Rodrigues da Silva, and Ricardo Pérez-Castillo, editors, *Quality of Information and Communications Technology*, pages 343–357, Cham, 2020. Springer International Publishing.

[BSN+09] Jan Brocke, Alexander Simons, Bjoern Niehaves, Björn Niehaves, Kai Riemer, Ralf Plattfaut, and Anne Cleven. Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process. *ECIS 2009 Proceedings*, January 2009.

[BvDV14] Eric Bouwers, Arie van Deursen, and Joost Visser. Towards a catalog format for software metrics. In *Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics*, WETSoM 2014, pages 44–47, New York, NY, USA, June 2014. Association for Computing Machinery.

[Cal05] Alan Calder. *IT Governance: Guidelines for Directors*. IT Governance Publishing, 2005.

[CC17] John W. Creswell and J. David Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications, December 2017.

[CH01] A. Cockburn and J. Highsmith. Agile software development, the people factor. *Computer*, 34(11):131–133, November 2001.

[CJS15] Yoon-Ho Choi, Han-You Jeong, and Seung-Woo Seo. A Quantitative Model for Evaluating the Efficiency of Proactive and Reactive Security Countermeasures. *IEICE Transactions on Information and Systems*, E98.D(3):637–648, 2015.

[Clo24] Google Cloud. Google security overview | Documentation. https://cloud.google.com/docs/security/overview/whitepaper, 2024. (Online, last accessed: 07.05.2024).

[Com24] European Comission. Cluster 3: Civil security for society (Horizon Europe) - European Commission. https://research-and-innovation.ec.europa.eu/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-europe/cluster-3-civil-security-society_en, April 2024. (Online, last accessed: 24.04.2024).

[Con09] Kieran Conboy. Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. *Information Systems Research*, 20(3):329–354, September 2009.

[Coo88] Harris M. Cooper. Organizing knowledge syntheses: A taxonomy of literature reviews. *Knowledge in Society*, 1(1):104, March 1988.

[Cou] PCI Security Standards Council. PCI Security Standards Council – Protect Payment Data with Industry-driven Security Standards, Training, and Programs. https://www.pcisecuritystandards.org/. (Online, last accessed: 25.04.2024).

[CRMO16] Marian Carcary, Karen Renaud, Stephen McLaughlin, and Conor O'Brien. A Framework for Information Security Governance and Management. *IT Professional*, 18(2):22–30, March 2016.

[Dam05] Marios Damianides. Sarbanes-Oxley and IT Governance: New Guidance on IT Control and Compliance. *IS Management*, 22:77–85, December 2005.

[DCB19] Sebastian Dännart, Fabiola Moyón Constante, and Kristian Beckers. An Assessment Model for Continuous Security Compliance in Large Scale Agile Environments: Exploratory Paper. In Paolo Giorgini and Barbara Weber, editors, *Advanced Information Systems Engineering*, volume 11483, pages 529–544, Cham, 2019. Springer International Publishing.

[DFI14] Torgeir Dingsøyr, Tor Erlend Fægri, and Juha Itkonen. What Is Large in Large-Scale? A Taxonomy of Scale for Agile Software Development. In Andreas Jedlitschka, Pasi Kuvaja, Marco Kuhrmann, Tomi Männistö, Jürgen Münch, and Mikko Raatikainen, editors, *Product-Focused Software Process Improvement*, pages 273–276, Cham, 2014. Springer International Publishing.

[DFP19] Torgeir Dingsoeyr, Davide Falessi, and Ken Power. Agile Development at Scale: The Next Frontier. *IEEE Software*, 36(2):30–38, March 2019.

[dig24] digital.ai. The 17th State of Agile Report. Technical report, 2024.

[DM13] Torgeir Dingsøyr and Nils Brede Moe. Research challenges in large-scale agile software development. *ACM SIGSOFT Software Engineering Notes*, 38(5):38–39, August 2013.

[DM14] Torgeir Dingsøyr and Nils Brede Moe. Towards Principles of Large-Scale Agile Development. In Torgeir Dingsøyr, Nils Brede Moe, Roberto Tonelli, Steve Counsell, Cigdem Gencel, and Kai Petersen, editors, *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*, pages 1–8, Cham, 2014. Springer International Publishing.

[DMFS18] Torgeir Dingsøyr, Nils Brede Moe, Tor Erlend Fægri, and Eva Amdahl Seim. Exploring software development at the very large-scale: A revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering*, 23(1):490–520, February 2018.

[DNBM12] Torgeir Dingsøyr, Sridhar Nerur, VenuGopal Balijepally, and Nils Brede Moe. A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, 85(6):1213–1221, June 2012.

[DPL16] Kim Dikert, Maria Paasivaara, and Casper Lassenius. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108, September 2016.

[DR13] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2013.

[DWA22] Lisa Nguyen Quang Do, James R. Wright, and Karim Ali. Why Do Software Developers Use Static Analysis Tools? A User-Centered Study of Developer Needs and Motivations. *IEEE Transactions on Software Engineering*, 48(3):835–847, March 2022.

[Eis89] Kathleen M. Eisenhardt. Building Theories from Case Study Research. *The Academy of Management Review*, 14(4):532–550, 1989.

[EP17] Christof Ebert and Maria Paasivaara. Scaling Agile. *IEEE Software*, 34(6):98–103, November 2017.

[Eur] European Union Agency for Cybersecurity (ENISA). ENISA Threat Landscape 2023. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023. (Online, last accessed: 25.04.2024).

[Eur16] European Union Agency for Cybersecurity (ENISA). Cyber Insurance: Recent Advances, Good Practices and Challenges. Report/Study, European Union, 2016.

[EWC21] Henry Edison, Xiaofeng Wang, and Kieran Conboy. Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 48(8):2709–2731, 2021.

[ext] Extreme Programming: A Gentle Introduction. http://www.extremeprogramming.org/. (Online, last accessed: 02.05.2024).

[Fed11] Federation of European Risk Management Associations (FERMA). Guidance on the 8th EU Company Law Directive. Technical report, 2011.

[FGJ14] Adila Firdaus, Imran Ghani, and Seung Ryul Jeong. Secure Feature Driven Development (SFDD) Model for Secure Software Development. *Procedia - Social and Behavioral Sciences*, 129:546–553, May 2014.

[Fin05] Arlene Fink. *Conducting Research Literature Reviews: From the Internet to Paper*. SAGE, 2005.

[FM91] Kevin Forsberg and Harold Mooz. The Relationship of System Engineering to the Project Cycle. *INCOSE International Symposium*, 1(1):57–65, 1991.

[Fow09] Floyd J. Fowler. *Survey Research Methods*. SAGE, 2009.

[FS17] Brian Fitzgerald and Klaas-Jan Stol. Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, 123:176–189, January 2017.

[FSOO13] Brian Fitzgerald, Klaas-Jan Stol, Ryan O'Sullivan, and Donal O'Brien. Scaling agile methods to regulated environments: An industry case study. In *2013 35th International Conference on Software Engineering (ICSE)*, pages 863–872, May 2013.

[FYLW20] Rubia Fatima, Affan Yasin, Lin Liu, and Jianmin Wang. The Use of Grey Literature and Google Scholar in Software Engineering Systematic Literature Reviews. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 1099–1100, July 2020.

[GAJ14] Imran Ghani, Zulkarnain Azham, and Seung Ryul Jeong. Integrating Software Security into Agile-Scrum Method. *KSII Transactions on Internet and Information Systems*, 8(2):646–663, February 2014.

[Ger] German Federal Office for Information Security. Cloud computing C5 criteria catalogue. https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Empfehlungen-nach-Angriffszielen/Cloud-Computing/Kriterienkatalog-C5/kriterienkatalog-c5.html?nn=909536. (Online, last accessed: 25.04.2024).

[GFM16] Vahid Garousi, Michael Felderer, and Mika V. Mäntylä. The need for multivocal literature reviews in software engineering: Complementing systematic literature reviews with grey literature. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, EASE '16, pages 1–6, New York, NY, USA, June 2016. Association for Computing Machinery.

[GFM19] Vahid Garousi, Michael Felderer, and Mika V. Mäntylä. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, 106:101–121, February 2019.

[Gil07] Asif Gill. Defining an Integrated Agile Governance for Large Agile Software Development Environments. In *Proceedings of the 8th Conference on Agile Processes in Software Engineering and Extreme Programming (XP2007)*, volume 4536, pages 157–160, June 2007.

[Gmb] VERBI GmbH. MAXQDA | All-In-One Qualitative & Mixed Methods Data Analysis Tool. https://www.maxqda.com/. (Online, last accessed: 07.05.2024).

[Gro24] BOC Group. ADOIT EA Suite. https://www.boc-group.com/en/adoit/, 2024. (Online, last accessed: 30.05.2024).

[Hal73] Edward T. Hall. *The Silent Language*. Knopf Doubleday Publishing Group, July 1973.

[Ham99] Booz Allen & Hamilton. Systems Security Engineering Capability Maturity Model SSE-CMM Model Description Document. Technical report, The SSE-CMM Project Community, 1999.

[Han10] Kanchan Hans. Cutting Edge Practices For Secure Software Engineering. *International Journal of Computer Science and Security*, 4, October 2010.

[HBSD18] Bettina Horlach, Tilo Böhmann, Ingrid Schirmer, and Paul Drews. IT Governance in Scaling Agile Frameworks. In *Tagungsband Multikonferenz Wirtschaftsinformatik*, Lüneburg, Germany, 2018.

[HHS+16] Geir K. Hanssen, Børge Haugset, Tor Stålhane, Thor Myklebust, and Ingar Kulbrandstad. Quality Assurance in Scrum Applied to Safety Critical Software. In Helen Sharp and Tracy Hall, editors, *Agile Processes, in Software Engineering, and Extreme Programming*, pages 92–103, Cham, 2016. Springer International Publishing.

[HMPR04] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design Science in Information Systems Research. *MIS Quarterly*, 28(1):75–105, 2004.

[HN18] Lise Tordrup Heeager and Peter Axel Nielsen. A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology*, 103:22–39, November 2018.

[HN19] Michael Huth and Flemming Nielson. Static Analysis for Proactive Security. In Bernhard Steffen and Gerhard Woeginger, editors, *Computing and Software Science*, volume 10000, pages 374–392, Cham, 2019. Springer International Publishing.

[HSG18] Rashina Hoda, Norsaremah Salleh, and John Grundy. The Rise and Evolution of Agile Software Development. *IEEE Software*, 35(5):58–63, September 2018.

[HSSH08] Mohammad Ali Hadavi, Hossein Shirazi, Hasan Mokhtari Sangchi, and Vahid Saber Hamishagi. Software Security; A Vulnerability Activity Revisit. In *2008 Third International Conference on Availability, Reliability and Security*, pages 866–872, March 2008.

[IBM24] IBM. Cost of a data breach 2023 | IBM. https://www.ibm.com/reports/data-breach, 2024. (Online, last accessed: 25.04.2024).

[ICO24] ICORE. ICORE Conference Rankings Portal. https://portal.core.edu.au/conf-ranks/, 2024. (Online, last accessed: 25.04.2024).

[Inc] TemboSocial Inc. Employee Recognition, Employee Feedback | TemboSocial. https://www.tembosocial.com. (Online, last accessed: 07.05.2024).

[Ins06] IT Governance Institute. *Information Security Governance: Guidance for Boards of Directors and Executive Management, 2nd Edition*. ISACA, 2006.

[Ins24] Project Management Institute. Foundation for Business Agility | Disciplined Agile®. https://www.pmi.org/disciplined-agile, 2024. (Online, last accessed: 26.04.2024).

[ISA12] ISACA. *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*. ISACA, 2012.

[ISOa] ISO/IEC. ISO/IEC 27000:2018. https://www.iso.org/standard/73906.html.

[ISOb] ISO/IEC. ISO/IEC 27001:2022. https://www.iso.org/standard/27001. (Online, last accessed: 25.04.2024).

[ISOc] ISO/IEC. ISO/IEC 38500:2024. https://www.iso.org/standard/81684.html. (Online, last accessed: 03.05.2024).

[ISO08] ISO/IEC. ISO/IEC 21827:2008(en), Information technology — Security techniques — Systems Security Engineering — Capability Maturity Model® (SSE-CMM®). https://www.iso.org/obp/ui/en/#iso:std:iso-iec:21827:ed-2:v1:en, 2008. (Online, last accessed: 25.04.2024).

[Jaj96] Sushil Jajodia. Managing security and privacy of information. *ACM Computing Surveys*, 28(4es):79–es, December 1996.

[JOKE+14] Alexandre J.H.de O.Luna, Philippe Kruchten, Marcello L.G. Do E.Pedrosa, Humberto R.de Almeida Neto, and Hermano P.de Moura Moura. State of the Art of Agile Governance: A Systematic Review. *International Journal of Computer Science and Information Technology*, 6(5):121–141, October 2014.

[JSMB13] Brittany Johnson, Yoonki Song, Emerson Murphy-Hill, and Robert Bowdidge. Why don't software developers use static analysis tools to find bugs? In *2013 35th International Conference on Software Engineering (ICSE)*, pages 672–681, May 2013.

[Jul08] Klaus Julisch. Security compliance: The next frontier in security research. In *Proceedings of the 2008 New Security Paradigms Workshop*, NSPW '08, pages 71–74, New York, NY, USA, September 2008. Association for Computing Machinery.

[KC07] Barbara Kitchenham and Stuart Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical report, Software Engineering Group School of Computer Science and Mathematics Keele University, Keele, Staffs, UK, January 2007.

[KHR18] Martin Kalenda, Petr Hyna, and Bruno Rossi. Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10):e1954, 2018.

[KI12] Henrik Kniberg and Anders Ivarsson. Scaling agile @ Spotify with tribes, squads, chapters & guilds, 2012.

[KKKI21] Rafiq Ahmad Khan, Siffat Ullah Khan, Habib Ullah Khan, and Muhammad Ilyas. Systematic Mapping Study on Security Approaches in Secure Software Engineering. *IEEE Access*, 9:19139–19160, 2021.

[KL20] Richard Knaster and Dean Leffingwell. *SAFe 5. 0 Distilled: Achieving Business Agility with the Scaled Agile Framework*. Addison-Wesley, 2020.

[KLHN18] Thomas Kude, Miroslav Lazic, Armin Heinzl, and Alexander Neff. Achieving IT-based synergies through regulation-oriented and consensus-oriented IT governance capabilities. *Information Systems Journal*, 28(5):765–795, 2018.

[KM08] Hossein Keramati and Seyed-Hassan Mirian-Hosseinabadi. Integrating software development security activities with agile methodologies. In *2008 IEEE/ACS International Conference on Computer Systems and Applications*, pages 749–754, March 2008.

[KNA+16] Raja Khaim, Saba Naz, Fakhar Abbas, Naila Iqbal, and Memoona Hamayun. A Review of Security Integration Technique in Agile Software Development. *International Journal of Software Engineering & Applications*, 7(3):49–68, May 2016.

[Kru04] Philippe Kruchten. Scaling down projects to meet the Agile sweet spot. *The Rational Edge*, January 2004.

[Kru13] Philippe Kruchten. Contextualizing agile software development. *Journal of Software: Evolution and Process*, 25(4):351–361, April 2013.

[KS21] Jan Koch and Carsten Schermuly. Who is attracted and why? How agile project management influences employee's attraction and commitment. *International Journal of Managing Projects in Business*, 14:699–720, April 2021.

[Kuc14] Udo Kuckartz. *Qualitative Text Analysis: A Guide to Methods, Practice & Using Software*. SAGE Publications Ltd, Los Angeles, 1 edition, January 2014.

[Kva07] Steinar Kvale. *Doing Interviews*. Sage Publications Ltd, Thousand Oaks, CA, 2007.

[KZ08] Muhammad Umair Ahmed Khan and Mohammad Zulkernine. Quantifying Security in Secure Software Development Phases. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*, pages 955–960, July 2008.

[KZ09] Muhammad Umair Ahmed Khan and Mohammed Zulkernine. On Selecting Appropriate Development Processes and Requirements Engineering Methods for Secure Software. In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, volume 2, pages 353–358, July 2009.

[LB18] Aurélie Leclercq-Vandelannoitte and Emmanuel Bertin. From sovereign IT governance to liberal IT governmentality? A Foucauldian analogy. *European Journal of Information Systems*, 27(3):326–346, May 2018.

[LCd+10] Alexandre J. H. de O. Luna, Cleyverson P. Costa, Hermano P. de Moura, Magdala A. Novaes, and Cesar A. D. C. do Nascimento. Agile governance in Information and Communication Technologies: Shifting paradigms. *JISTEM Journal of Information Systems and Technology Management*, 7(2):311–334, August 2010.

[LEP14] Roman Lukyanenko, Joerg Evermann, and Jeffrey Parsons. Instantiation Validity in IS Design Research. In Monica Chiarini Tremblay, Debra VanderMeer, Marcus Rothenberger, Ashish Gupta, and Victoria Yoon, editors, *Advancing the Impact of Design Science: Moving from Theory to Practice*, pages 321–328, Cham, 2014. Springer International Publishing.

[LJE06] Yair Levy and Timothy J. Ellis. A Systems Approach to Conduct an Effective Literature Review in Support of Information Systems Research. *Informing Science: The International Journal of an Emerging Transdiscipline*, 9:181–212, 2006.

[LKRM16] Alexandre Luna, Philippe Kruchten, Edson Riccio, and Hermano Moura. Foundations for an Agile Governance Manifesto: A bridge for business agility. In *Proceedings of the 13th International Conference on Management of Technology and Information Systems*, São Paulo, SP, Brazil, June 2016.

[LMd20] Alexandre J. H. de O. Luna, Marcelo L. M. Marinho, and Hermano P. de Moura. Agile governance theory: Operationalization. *Innovations in Systems and Software Engineering*, 16(1):3–44, March 2020.

[LSC20] Martin Forsberg Lie, Mary Sánchez-Gordón, and Ricardo Colomo-Palacios. DevOps in an ISO 13485 Regulated Environment: A Multivocal Literature Review. In *Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, ESEM '20, pages 1–11, New York, NY, USA, October 2020. Association for Computing Machinery.

[LV16] Craig Larman and Bas Vodde. *Large-Scale Scrum: More with LeSS*. Addison-Wesley Professional, 1 edition, September 2016.

[MAR+20] Fabiola Moyon, Pamela Almeida, Daniel Riofrio, Daniel Mendez, and Marcos Kalinowski. Security Compliance in Agile Software Development: A Systematic Mapping Study. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 413–420, Portoroz, Slovenia, August 2020. IEEE.

[May00] Philipp Mayring. Qualitative Content Analysis. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research [On-line Journal], http://qualitative-research.net/fqs/fqs-e/2-00inhalt-e.htm*, 1(2), June 2000.

[MBK+18] Fabiola Moyon, Kristian Beckers, Sebastian Klepper, Philipp Lachberger, and Bernd Bruegge. Towards continuous security compliance in agile software development at scale. In *Proceedings of the 4th International Workshop on Rapid Continuous*

Software Engineering, RCoSE '18, pages 31–34, New York, NY, USA, May 2018. Association for Computing Machinery.

[MC16] John R. Michener and Aaron T. Clager. Mitigating an Oxymoron: Compliance in a DevOps Environments. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 396–398, June 2016.

[MC17] Håvard Myrbakken and Ricardo Colomo-Palacios. DevSecOps: A Multivocal Literature Review. In Antonia Mas, Antoni Mesquida, Rory V. O'Connor, Terry Rout, and Alec Dorling, editors, *Software Process Improvement and Capability Determination*, pages 17–29, Cham, 2017. Springer International Publishing.

[McG02] G. McGraw. Building secure software: Better than protecting bad software. *IEEE Software*, 19(6):57–58, November 2002.

[McG03] G. McGraw. From the ground up: The DIMACS software security workshop. *IEEE Security & Privacy*, 1(2):59–66, March 2003.

[McG04] G. McGraw. Software security. *IEEE Security & Privacy*, 2(2):80–83, March 2004.

[McG06a] Gary McGraw. Software Security: Building Security In. In *2006 17th International Symposium on Software Reliability Engineering*, pages 6–6, November 2006.

[McG06b] Gary McGraw. *Software Security: Building Security In*. Addison-Wesley Professional, 2006.

[MHS13] Matthew B. Miles, A. Michael Huberman, and Johnny Saldana. *Qualitative Data Analysis: A Methods Sourcebook*. SAGE Publications, April 2013.

[Mic] Microsoft. Microsoft Security Development Lifecycle. https://www.microsoft.com/en-us/securityengineering/sdl. (Online, last accessed: 03.05.2024).

[Mic16] Microsoft. Security for Modern Engineering. Technical report, Microsoft, 2016.

[MK11] Michael D. Myers and Heinz K. Klein. A Set of Principles for Conducting Critical Research in Information Systems. *MIS Quarterly*, 35(1):17–36, 2011.

[MMBK21] Fabiola Moyón, Daniel Méndez, Kristian Beckers, and Sebastian Klepper. How to Integrate Security Compliance Requirements with Agile Software Engineering at Scale? In Maurizio Morisio, Marco Torchiano, and Andreas Jedlitschka, editors, *Product-Focused Software Process Improvement*, volume 12562, pages 69–87, Cham, 2021. Springer International Publishing.

[MN07] Michael D. Myers and Michael Newman. The qualitative interview in IS research: Examining the craft. *Information and Organization*, 17(1):2–26, January 2007.

[MS16] Zhendong Ma and Christoph Schmittner. Threat Modeling for Automotive Security Analysis. In *Proceedings of the the 9th International Conference on Security Technology*, pages 333–339, November 2016.

[MŠPL21] Nils Brede Moe, Darja Šmite, Maria Paasivaara, and Casper Lassenius. Finding the sweet spot for organizational control and team autonomy in large-scale agile software development. *Empirical Software Engineering*, 26(5):101, September 2021.

[Mye13] Michael D. Myers. *Qualitative Research in Business and Management.* SAGE Publications, April 2013.

[MZD+20] Runfeng Mao, He Zhang, Qiming Dai, Huang Huang, Guoping Rong, Haifeng Shen, Lianping Chen, and Kaixiang Lu. Preliminary Findings about DevSecOps from Grey Literature. In *2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*, pages 450–457, December 2020.

[NAD20] Nathan Newton, Craig Anslow, and Andreas Drechsler. Information Security in Agile Software Development Projects: A Critical Success Factor Perspective. In *Proceedings of the 27th Conference on Information Systems (ECIS)*, Uppsala, Sweden, October 2020.

[NR21] Rennie Naidoo and Sydwell Rikhotso. Balancing Autonomy and Control Tensions in Large-Scale Agile. *Proceedings of the International Conference on Information Systems Development (ISD)*, August 2021.

[NSBJ10] Torstein Nicolaysen, Richard Sasson, Maria Bartnes, and Martin Jaatun. Agile Software Development: The Straight and Narrow Path to Secure Software? *IJSSE*, 1:71–85, July 2010.

[OCJ16] Tosin Daniel Oyetoyan, Daniela Soares Cruzes, and Martin Gilje Jaatun. An Empirical Study on the Relationship between Software Security Skills, Usage and Training Needs in Agile Settings. In *2016 11th International Conference on Availability, Reliability and Security (ARES)*, pages 548–555, August 2016.

[ORbO15] Hela Oueslati, Mohammad Masudur Rahman, and Lotfi ben Othmane. Literature Review of the Challenges of Developing Secure Software Using the Agile Approach. In *2015 10th International Conference on Availability, Reliability and Security*, pages 540–547, August 2015.

[OV] OWASP and Robert Van der Veer. SAMM Agile Guidance. https://owaspsamm.org/guidance/agile/. (Online, last accessed: 07.05.2024).

[OWAa] OWASP. OWASP Devsecops Maturity Model | OWASP Foundation. https://owasp.org/www-project-devsecops-maturity-model/. (Online, last accessed: 26.04.2024).

[OWAb] OWASP. OWASP SAMM | OWASP Foundation. https://owasp.org/www-project-samm/. (Online, last accessed: 26.04.2024).

[PC] European Parliament and Council of the European Union. Art. 32 GDPR – Security of processing.

[PF01] Steve R. Palmer and Mac Felsing. *A Practical Guide to Feature-Driven Development.* Pearson Education, 1st edition, October 2001.

[PJR20] Alexander Poth, Jan Jacobsen, and Andreas Riel. Systematic Agile Development in Regulated Environments. In Murat Yilmaz, Jörg Niemann, Paul Clarke, and Richard Messnarz, editors, *Systems, Software and Services Process Improvement*, pages 191–202, Cham, 2020. Springer International Publishing.

[PKHR21] Alexander Poth, Mario Kottke, Christian Heimann, and Andreas Riel. The EFIS Framework for Leveraging Agile Organizations Within Large Enterprises. In Peggy Gregory and Philippe Kruchten, editors, *Agile Processes in Software Engineering and Extreme Programming – Workshops*, volume 426, pages 42–51, Cham, 2021. Springer International Publishing.

[PKM+21] Alexander Poth, Mario Kottke, Kerstin Middelhauve, Torsten Mahr, and Andreas Riel. Lean integration of IT security and data privacy governance aspects into product development in agile organizations. *JUCS - Journal of Universal Computer Science*, 27(8):868–893, August 2021.

[PKMR23] Alexander Poth, Mario Kottke, Torsten Mahr, and Andreas Riel. Teamwork quality in technology-driven product teams in large-scale agile organizations. *Journal of Software: Evolution and Process*, 35(8):e2388, 2023.

[PKR21] Alexander Poth, Mario Kottke, and Andreas Riel. Measuring teamwork quality in large-scale agile organisations evaluation and integration of the aTWQ approach. *IET Software*, 15, August 2021.

[PKT+17] Andreas Poller, Laura Kocksch, Sven Türpe, Felix Anand Epp, and Katharina Kinder-Kurlanda. Can Security Become a Routine? A Study of Organizational Change in an Agile Software Development Group. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, pages 2489–2503, New York, NY, USA, February 2017. Association for Computing Machinery.

[PM12] R Pereira and Miguel Mira da Silva. A Literature Review: Guidelines and Contingency Factors for IT Governance. In *Proceedings of the European, Mediterranean & Middle Eastern Conference on Information Systems 2012 (EMOIS2012)*, Munich, Germany, January 2012. Emerald Group Publishing Limited.

[PM19] Yvan Petit and Carl Marnewick. Earn Your Wings: A Novel Approach to Deployment Governance. In Rashina Hoda, editor, *Agile Processes in Software Engineering and Extreme Programming – Workshops*, pages 64–71, Cham, 2019. Springer International Publishing.

[PPL18] Abheeshta Putta, Maria Paasivaara, and Casper Lassenius. Benefits and Challenges of Adopting the Scaled Agile Framework (SAFe): Preliminary Results from a Multivocal Literature Review. In Marco Kuhrmann, Kurt Schneider, Dietmar Pfahl, Sousuke Amasaki, Marcus Ciolkowski, Regina Hebig, Paolo Tell, Jil Klünder, and Steffen Küpper, editors, *Product-Focused Software Process Improvement*, pages 334–351, Cham, 2018. Springer International Publishing.

[PQS24] Cherilyn Pascoe, Stephen Quinn, and Karen Scarfone. The NIST Cybersecurity Framework (CSF) 2.0. *NIST*, February 2024.

[PRTV12] Ken Peffers, Marcus Rothenberger, Tuure Tuunanen, and Reza Vaezi. Design Science Research Evaluation. In Ken Peffers, Marcus Rothenberger, and Bill Kuechler, editors, *Design Science Research in Information Systems. Advances in Theory and Practice*, pages 398–410, Berlin, Heidelberg, 2012. Springer.

[PTRC07] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and Samir Chatterjee. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3):45–77, December 2007.

[Pv04] Shaun Posthumus and Rossouw von Solms. A framework for the governance of information security. *Computers & Security*, 23(8):638–646, December 2004.

[PW10] Kai Petersen and Claes Wohlin. The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering*, 15(6):654–693, December 2010.

[PWB09] Kai Petersen, Claes Wohlin, and Dejan Baca. The Waterfall Model in Large-Scale Development. In Frank Bomarius, Markku Oivo, Päivi Jaring, and Pekka Abrahamsson, editors, *Product-Focused Software Process Improvement*, pages 386–400, Berlin, Heidelberg, 2009. Springer.

[Qua24] Qualis. Qualis CAPES Conference Rankings. https://ppgcc.github.io/discentesPPGCC/en/qualis/, 2024. (Online, last accessed: 24.04.2024).

[RH09] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, April 2009.

[RHA+18] Klaus Reche Riisom, Martin Slusarczyk Hubel, Hasan Mousa Alradhi, Niels Bonde Nielsen, Kati Kuusinen, and Ronald Jabangwe. Software security in agile software development: A literature review of challenges and solutions. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, XP '18, pages 1–5, New York, NY, USA, May 2018. Association for Computing Machinery.

[RHL15] Kalle Rindell, Sami Hyrynsalmi, and Ville Leppänen. A comparison of security assurance support of agile software development methods. In *Proceedings of the 16th International Conference on Computer Systems and Technologies*, CompSysTech '15, pages 61–68, New York, NY, USA, June 2015. Association for Computing Machinery.

[RHL18] Kalle Rindell, Sami Hyrynsalmi, and Ville Leppänen. Aligning security objectives with agile software development. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, XP '18, pages 1–9, New York, NY, USA, May 2018. Association for Computing Machinery.

[RR12] Herbert J. Rubin and Irene S. Rubin. *Qualitative Interviewing: The Art of Hearing Data*. SAGE Publications, 2012.

[RRH+21] Kalle Rindell, Jukka Ruohonen, Johannes Holvitie, Sami Hyrynsalmi, and Ville Leppänen. Security in agile software development: A practitioner survey. *Information and Software Technology*, 131:106488, March 2021.

[RRS+20] Harvard Business Review, Darrell K. Rigby, Jeff Sutherland, Peter Cappelli, and Phil Simon. *Agile: Tools for Preparing Your Team for the Future (The Insights You Need from Harvard Business Review)*. Harvard Business Review Press, Boston, Massachusetts, April 2020.

[RZ22] Sarah Rietze and Hannes Zacher. Relationships between Agile Work Practices and Occupational Well-Being: The Role of Job Demands and Resources. *International Journal of Environmental Research and Public Health*, 19:1258, January 2022.

[SA24a] Inc Scaled Agile. SAFe Principle #6 - Make Value Flow Without Interruptions. https://scaledagileframework.com/make-value-flow-without-interruptions/, 2024. (Online, last accessed: 26.04.2024).

[SA24b] Inc Scaled Agile. The Scaled Agile Framework (SAFe) 6.0. https://scaledagileframework.com/, 2024. (Online, last accessed: 26.04.2024).

[Sal15] Johnny Saldaña. *The Coding Manual for Qualitative Researchers*. SAGE Publications Ltd, Los Angeles, Calif. London New Delhi Singapore Washington DC, 3rd edition edition, December 2015.

[SAP20] SAP. The Secure Software Development Lifecycle at SAP. Technical report, SAP, 2020.

[SBK05] M. Siponen, R. Baskerville, and T. Kuivalainen. Integrating Security into Agile Development Methods. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pages 185a–185a, January 2005.

[SC90] Anselm Strauss and Juliet M. Corbin. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. Basics of Qualitative Research: Grounded Theory Procedures and Techniques. Sage Publications, Inc, Thousand Oaks, CA, US, 1990.

[SC14] Spyridon Samonas and David Coss. The CIA Strikes Back: Redefining Confidentiality, Integrity and Availability in Security. *Journal of Information Systems Security*, 10(3):21–45, 2014.

[Scr] What is Scrum? | Scrum.org. https://www.scrum.org/resources/what-scrum-module. (Online, last accessed: 02.05.2024).

[Sea99] C.B. Seaman. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on Software Engineering*, 25(4):557–572, July-Aug./1999.

[Sho14] Adam Shostack. *Threat Modeling: Designing for Security*. Wiley, Indianapolis, IN, 1st edition, February 2014.

[SIG24] SIGSOFT. ACM SIGSOFT Empirical Standards for Software Engineering. ACM Special Interest Group on Software Engineering, April 2024.

[SJD19] Daniel Sawano, Dan Bergh Johnsson, and Daniel Deogun. *Secure by Design*. Simon and Schuster, September 2019.

[SKHW19] Jan-Philipp Steghöfer, Eric Knauss, Jennifer Horkoff, and Rebekka Wohlrab. Challenges of Scaled Agile for Safety-Critical Systems. In *Proceedings of Product-Focused Software Process Improvement (Profes)*, volume 11915, pages 350–366, 2019.

[SMH18] Viktoria Stray, Nils Brede Moe, and Rashina Hoda. Autonomous agile teams: Challenges and future directions for research. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pages 1–5, May 2018.

[Sol98] R. V. Solms. Information Management & Computer Security Information security management (3) : The Code of Practice for Information Security Management (BS 7799). *Information Management & Computer Security,*, 6(4), 1998.

[Son24] SonarSource. Code Quality Tool & Secure Analysis with SonarQube. https://www.sonarsource.com/products/sonarqube/, 2024. (Online, last accessed: 30.05.2024).

[SPD21] Neha Singh, Palak Patel, and Soma Datta. A survey on security and human-related challenges in agile software deployment. In *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1976–1982, December 2021.

[SPL+23] Irina Safonova, Maria Paasivaara, Casper Lassenius, Ömer Uludağ, and Abheeshta Putta. Experienced Challenges of Adopting Agile Scaling Frameworks. In *2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–10, October 2023.

[SS] Ken Schwaber and Scrum.org. The Nexus™ Guide | Scrum.org. https://www.scrum.org/resources/nexus-guide. (Online, last accessed: 07.05.2024).

[SS09] S. H. Solms and Rossouw Solms. *Information Security Governance*. Springer New York, NY, 2009.

[SS20] Stef Schinagl and Abbas Shahim. What do we know about information security governance? "From the basement to the boardroom": Towards digital security governance. *Information & Computer Security*, 28(2):261–292, January 2020.

[SSC+18] Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko, and Alberto Bacchelli. Modern code review: A case study at google. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, ICSE-SEIP '18, pages 181–190, New York, NY, USA, May 2018. Association for Computing Machinery.

[Sul19] Bryan Sullivan. Agile SDL: Streamline Security Practices For Agile Development. https://learn.microsoft.com/en-us/archive/msdn-magazine/2008/november/agile-sdl-streamline-security-practices-for-agile-development, September 2019. (Online, last accessed: 26.04.2024).

[Sut] Jeff Sutherland. The Official Scrum@Scale Guide | Scrum@Scale Framework. https://www.scrumatscale.com/scrum-at-scale-guide/. (Online, last accessed: 02.05.2024).

[Syn] Synopsys. Building Security Maturity Model (BSIMM) Consulting Services | Synopsys. https://www.synopsys.com/software-integrity/software-security-services/bsimm-maturity-model.html. (Online, last accessed: 26.04.2024).

[syn23] synopsys. 14th BSIMM Software Security Assessment Report | Synopsys. https://www.synopsys.com/software-integrity/resources/analyst-reports/bsimm.html, 2023. (Online, last accessed: 26.04.2024).

[Tas09] Igli Tashi. Regulatory Compliance and Information Security Assurance. In *2009 International Conference on Availability, Reliability and Security*, pages 670–674, March 2009.

[TD09] David Talby and Yael Dubinsky. Governance of an agile software project. In *2009 ICSE Workshop on Software Development Governance*, pages 40–45, May 2009.

[TP17] Sven Türpe and Andreas Poller. Managing Security Work in Scrum: Tensions and Challenges. In *Proceedings of the International Workshop on Secure Software Engineering in DevOps and Agile Development*, 2017.

[UKXM17] Ömer Uludağ, Martin Kleehaus, Xian Xu, and Florian Matthes. Investigating the Role of Architects in Scaling Agile Frameworks. In *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*, pages 123–132, October 2017.

[UNH19] Ömer Uludag, Sascha Nägele, and Matheus Hauder. Establishing Architecture Guidelines in Large-Scale Agile Development Through Institutional Pressures: A Single-Case Study. In *AMCIS 2019 Proceedings*, July 2019.

[UNHM21] Ömer Uludağ, Sascha Nägele, Matheus Hauder, and Florian Matthes. A Tool Supporting Architecture Principles and Guidelines in Large-Scale Agile Development. In Alfred Zimmermann, Rainer Schmidt, and Lakhmi C. Jain, editors, *Architecting the Digital Transformation: Digital Business, Technology, Decision Support, Management*, pages 327–344. Springer International Publishing, Cham, 2021.

[UPP+22] Ömer Uludağ, Pascal Philipp, Abheeshta Putta, Maria Paasivaara, Casper Lassenius, and Florian Matthes. Revealing the state of the art of large-scale agile development research: A systematic mapping study. *Journal of Systems and Software*, 194:111473, December 2022.

[UPPM21] Ömer Uludağ, Abheeshta Putta, Maria Paasivaara, and Florian Matthes. Evolution of the Agile Scaling Frameworks. In Peggy Gregory, Casper Lassenius, Xiaofeng Wang, and Philippe Kruchten, editors, *Agile Processes in Software Engineering and Extreme Programming*, pages 123–139, Cham, 2021. Springer International Publishing.

[vBS18] Amber van der Heijden, Cosmin Broasca, and Alexander Serebrenik. An empirical perspective on security challenges in large-scale agile software development. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '18, pages 1–4, New York, NY, USA, October 2018. Association for Computing Machinery.

[VDA] VDA. VDA ISA catalog. https://www.vda.de/en/topics/digitization/data/information-security. (Online, last accessed: 25.04.2024).

[vDHG06] Wim van grembergen, Steven De Haes, and Erik Guldentops. Structures, Processes and Relational Mechanisms for IT Governance. In *Strategies for Information Technology Governance*. Idea Group Publishing, January 2006.

[vKd23] Robert M. van Wessel, Philip Kroon, and Henk J. de Vries. Scaling Agile Company Wide: The Organizational Challenge of Combining Agile Scaling Frameworks and Enterprise Architecture in Service Companies. *IEEE Engineering Management Review*, 51(3):25–32, 2023.

[VM01] J Viega and Gary McGraw. *Building Secure Software: How to Avoid Security Problems the Right Way*. Addison-Wesley Longman Publishing Co., Inc., USA, September 2001.

[von05] Basie von Solms. Information Security Governance – Compliance management vs operational management. *Computers & Security*, 24(6):443–447, September 2005.

[VRC19] Sulejman Vejseli, Alexander Rossmann, and Thomas Connolly. IT Governance and Its Agile Dimensions. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, January 2019.

[VRC20] Sulejman Vejseli, Alexander Rossmann, and Thomas Connolly. Agility matters! Agile Mechanisms in IT Governance and their Impact on Firm Performance. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 53rd Hawaii International Conference on System Sciences, January 2020.

[VS01] Basie Von Solms. Corporate Governance and Information Security. *Computers and Security*, 20(3):215–218, May 2001.

[VS06] Basie Von Solms. Information Security - The Fourth Wave. *Computers and Security*, 25(3):165–168, May 2006.

[vv13] Rossouw von Solms and Johan van Niekerk. From information security to cyber security. *Computers & Security*, 38:97–102, October 2013.

[WC03] L. Williams and A. Cockburn. Agile software development: It's about feedback and change. *Computer*, 36(6):39–43, June 2003.

[Wei06] Peter Weill. The Agility Paradox, 2006.

[Wei08] Peter Weill. Don't Just Lead, Govern: How Top-Performing Firms Govern IT. *MIS Quarterly Executive*, 3(1), February 2008.

[Wil01] Paul Williams. Information Security Governance. *Information Security Technical Report*, 6(3):60–70, September 2001.

[WMW22] Charles Weir, Sammy Migues, and Laurie Williams. Exploring the Shift in Security Responsibility. *IEEE Security and Privacy*, 20(6):8–17, November 2022.

[WR04] Peter Weill and Jeanne Ross. *IT Governance: How Top Performers Manage IT Decision Rights for Superior Results*. Harvard Business Press, June 2004.

[WRH+22] Diana Marie Wiechmann, Christopher Reichstein, Ralf-Christian Haerting, Joerg Bueechl, and Michael Pressl. Agile management to secure competitiveness in times of digital transformation in medium-sized businesses. *Procedia Computer Science*, 207:2353–2363, January 2022.

[Wri14] Christopher Wright. *Agile Governance and Audit: An Overview for Auditors and Agile Teams*. IT Governance Publishing, 2014.

[WW02] Jane Webster and Richard T. Watson. Analyzing the Past to Prepare for the Future: Writing a Literature Review. *MIS Quarterly*, 26(2):xiii–xxiii, 2002.

[Yil13] Kaya Yilmaz. Comparison of Quantitative and Qualitative Research Traditions: Epistemological, theoretical, and methodological differences. *European Journal of Education*, 48(2):311–325, 2013.

[Yin03] Robert K. Yin. *Case Study Research: Design and Methods*. SAGE, 2003.

[ZJZ+16] Xin Zhou, Yuqin Jin, He Zhang, Shanshan Li, and Xin Huang. A Map of Threats to Validity of Systematic Literature Reviews in Software Engineering. In *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*, pages 153–160, December 2016.

Publications

[NKM23] Sascha Nägele, Lorena Korn, and Florian Matthes. Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*, ARES '23, pages 1–12, New York, NY, USA, August 2023. Association for Computing Machinery.

[NSFM24] Sascha Nägele, Nathalie Schenk, Nico Fechtner, and Florian Matthes. Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development. In *Proceedings of the 26th International Conference on Enterprise Information Systems*, Angers, France, 2024. SCITEPRESS – Science and Technology Publications, Lda.

[NSM23] Sascha Nägele, Nathalie Schenk, and Florian Matthes. The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study. In *2023 IEEE 25th Conference on Business Informatics (CBI)*, pages 1–10, June 2023.

[NWM22] Sascha Nägele, Jan-Philipp Watzelt, and Florian Matthes. Investigating the Current State of Security in Large-Scale Agile Development. In Viktoria Stray, Klaas-Jan Stol, Maria Paasivaara, and Philippe Kruchten, editors, *Agile Processes in Software Engineering and Extreme Programming*, pages 203–219, Cham, 2022. Springer International Publishing.

[NWM24] Sascha Nägele, Jan-Philipp Watzelt, and Florian Matthes. Assessing Team Security Maturity in Large-Scale Agile Development. In *Hawaiian International Conference of System Sciences (HICSS)*, January 2024.

Abbreviations

ASD	Agile Software Development
BSIMM	Building Security In Maturity Model
DAST	Dynamic Application Security Testing
DA	Disciplined Agile
DoD	Definition of Done
DS	Design Science
DSR	Design Science Research
DSOMM	DevSecOps Maturity Model
LeSS	Large Scale Scrum
LoD	Lines of Defense
LSAD	Large-Scale Agile Development
OWASP	Open Web Application Security Project
RQ	Research Question
SAFe	Scaled Agile Framework
SSE	Secure Software Engineering
SAMM	Software Assurance Maturity Model
SAST	Static Application Security Testing

SC	Security Champion
SDL	Secure Development Lifecycle
SDLC	Software Development Lifecycle
SE	Security Engineer
SLR	Systematic Literature Review
SSE-CMM	System Security Engineering Capability Maturity Model
TSMM	Team Security Maturity Model

APPENDIX A

Embedded Publications in Original Format

Investigating the Current State of Security in Large-Scale Agile Development

Sascha Nägele( ), Jan-Philipp Watzelt, and Florian Matthes

Technical University of Munich, Munich, Germany
{sascha.naegele,jan-philipp.watzelt,matthes}@tum.de

Abstract. Agile methods have become the established way to successfully handle changing requirements and time-to-market pressure, even in large-scale environments. Simultaneously, security has become an increasingly important concern due to more frequent and impactful incidents, stricter regulations with growing fines, and reputational damages. Despite its importance, research on how to address security in large-scale agile development is scarce. Therefore, this paper provides an empirical investigation on tackling software product security in large-scale agile environments. Based on a literature review and preliminary interviews, we identified four essential categories that impact how to handle security: (i) the structure of the agile program, (ii) security governance, (iii) adaptions of security activities to agile processes, and (iv) tool-support and automation. We conducted semi-structured interviews with nine experts from nine companies in five industries based on these categories. We performed a content-structuring qualitative analysis to reveal recurring patterns of best practices and challenges in those categories and identify differences between organizations. Among the key findings is that the analyzed organizations introduce cross-team security-focused roles collaborating with agile teams and use automation where possible. Moreover, security governance is still driven top-down, which conflicts with team autonomy in agile settings.

Keywords: Large-scale agile · Security · Software development

1 Introduction

The use of agile methods is omnipresent. According to the most recent "State of Agile Report", agile adoption within software development teams has surged from 37% in 2020 to 86% in 2021 [11]. Agile development methods are also increasingly applied to large projects and companies with numerous software development teams working together [12]. Companies thereby aim to benefit from the advantages of these methods, such as enhanced adaptability to fast-evolving environments and accelerated time-to-market [37].

V. Stray et al. (Eds.): XP 2022, LNBIP 445, pp. 203–219, 2022.
https://doi.org/10.1007/978-3-031-08169-9_13

At the same time, software security is becoming an increasingly important concern due to stricter legislation and growing fines [9]. In addition, there is a growing intrinsic motivation for companies to pay more attention to security. As a global risk management survey with thousands of participating companies shows, cyberattacks, data breaches, and reputational damage are the most significant perceived risks to business success [4]. The global Covid-19 pandemic further exacerbates the complexity and growing number of cyberattacks as changing work conditions and consumer behavior further increase the dependence on Information Technology (IT) [14]. Despite the importance of software security in scaled agile environments, there are only few empirical studies, and more empirical research is needed [1,31,48].

This study contributes to the empirical evidence on how organizations tackle software security in large-scale agile development (LSAD). The primary research question we strive to answer is: **How is security approached in LSAD, and what are recurring best practices and challenges?** We provide a cross-industry overview based on literature and interviews with nine experts from nine companies in five industries. The remainder of this paper is organized as follows: Section 2 presents the theoretical background and related work. Section 3 explains the research methodology. Section 4 summarizes the results, which are discussed in Sect. 5. Section 6 presents the conclusion and outlook.

2 Background and Related Work

We follow Dikert et al. in defining LSAD, who speak of a minimum of 50 people or at least six teams [12].

One of the earlier related works is by Bartsch [45], who studied security in agile development by interviewing ten practitioners but does not explicitly address LSAD contexts yet. Relevant for our work is the more recent study by Amber et al. who identified three unique security challenges in LSAD: "(i) alignment of security objectives in a distributed setting; (ii) developing a common understanding of roles and responsibilities in security activities; and (iii) integration of low-overhead security testing tools" [48]. Our key findings discuss how our results relate to these challenges.

In addition, valuable related work includes widespread software security maturity frameworks, e.g., the Building Security in Maturity Model (BSIMM) [28] and the OWASP Software Assurance Maturity Model (SAMM) [35]. These are mainly driven by practical experience from the industry and provide a highly comprehensive insight into secure software development initiatives. Even if they do not explicitly address LSAD and describe themselves as agnostic of the development approach, many of the listed organizations working with these models fulfill the definition of LSAD. However, we base our study on a literature review to achieve unbiased research independent of these models.

In the following subsections, we present the theoretical background and related work using four categories that emerged from our literature review and can be mapped to Amber et al.'s [48] challenges. We also use these categories to structure our interviews and results.

Structure of the Agile Program. Poller et al. [38] emphasize that considering organizational structures, e.g., roles and their interaction, is vital for promoting security approaches and governing agile teams in LSAD. Alsaqaf et al. [1] found in a systematic literature review that additional roles are introduced in LSAD to address quality requirements, e.g., a security architect. The authors emphasize that further empirical research on such roles is needed. Newton et al. [33] discovered security-related communities of practices, while Rindell et al. [39] observed an internal software security group that, e.g., carries out security reviews. Steghöfer et al. and Dännart et al. note that LSAD frameworks do not provide security compliance out-of-the-box [10,46]. Moyon et al. [30] recommend further adaptions, e.g., by introducing security roles. Oyetoyan et al. [36] describe a group of security experts supporting with, e.g., adherence to security standards and organizing security audits. The proposal of Boström et al. [8] includes a team of security engineers, e.g., to support the definition of security stories and risk assessments together with product teams.

Also, publications from software companies such as SAP [42], Microsoft [27] and Google [15] show that dedicated security roles are being used in practice, although the exact range of tasks is not always explained in detail. We thereby derive that the *structure of the agile program* is vital for addressing security in LSAD.

Security Governance. Security governance can be seen as a subset of IT governance, often characterized by top-down control [17]. Despite limited empirical studies on IT governance in agile and lean environments, its importance has been recognized [47]. The literature recommends moving to agile and lean governance approaches to better align governance and agility. The term lean governance is more frequently used in industry publications such as white papers and large-scale agile frameworks [3,43]. Horlach et al. [16] found that traditional governance structures hinder autonomous agile teams in LSAD. Ambler [2] stated early on that a lean form of IT governance is required to achieve agility in software development at scale. Vejseli et al. [49] found that agile IT governance positively affects business-IT alignment and, thus, enterprise performance, similar to traditional governance. By fostering the necessary engagement of all parts of the business, agile governance helps increase business agility [23]. Agile governance focuses on enabling and motivating development teams through collaborative and supportive practices [2]. Instead of top-down control, it promotes bottom-up engagement, autonomy, and self-organization [3,24]. Because of this tension, we derive *security governance* as an essential category.

Security Activities. We understand security activities as a set of practices that directly or indirectly enhance software security. A typical example is threat modeling. It is a component of security risk analysis [25] and supports the identification of security risks and appropriate measures [44]. Other common examples are penetration testing [5] and code reviews [41]. Multiple researchers agree that incorporating security activities in agile development is feasible and necessary [31,33]. Beznosov and Kruchten [7] propose integration strategies depending on the match between security practices and agile principles. As stated by Keramati

and Mirian-Hosseinabadi, security activities are integrated with agile software development based on balancing "the costs of decreased level of agility [...] and benefits from developing more secure systems" [19]. Hence, we derive the category of *security activities* for our interviews.

Tool-Support and Automation. In their case study, Barbosa and Sampaio note that the "demand to build software quickly and cost-effectively" impedes the integration of agile security approaches due to the associated cost and time effort [6]. Therefore, automating manual, work-intensive tasks is crucial to reduce the friction between security and iterative deployment practices. In recent years, the term DevSecOps matured from a buzzword to a well-established movement [32]. One of the primary goals is integrating security activities and practices into development pipelines facilitated by security automation tools [29]. Researchers emphasize automating repetitive manual tasks, like security code reviews, to ensure security while sustaining a high velocity in agile software development [18,34]. Examples of security automation include static and dynamic application security testing. Since reducing manual effort and a more frictionless integration of security activities is critical in scaled environments, we derive *tool-support and automation* as the fourth category for our interviews.

3 Research Methodology

We present the three stages of our methodological process below: study design, data collection, and analysis.

Study Design. To gain cross-case insights into our research question, we deemed an interview study the most suitable primary research method. We excluded a multiple case study because not enough cases provided multiple sources for data collection due to the topic's sensitive nature. To allow for a better aggregation and comparability of results, we roughly structured the interview with the categorization described in the background and related work. Before conducting the actual interview study, we performed four preliminary expert interviews in two organizations to discuss and evaluate the categorization. In each category, we used semi-structured questions, which allow for enough freedom in the answers and the possibility for individual adjustments during the interview [13].

In contrast to expert-focused surveys, we also considered the experts' current organizations, i.e., we did not select the experts solely based on their role, competency, and experience, but an important factor was the organization they currently work for. The organizations must fulfill the previously described definition of LSAD.

Data Collection. For the interview study data collection, experts from nine companies participated in our study.

We collected data across five industries to ensure better generalizability of results. The following sectors are represented based on the main product focus

of the case company: IT and software development, software development consulting, media, insurance, and automotive. Two researchers interviewed six of the nine interviewees. Three were interviewed by one researcher. After obtaining explicit consent to record the interviews for transcription purposes, we used online video conferencing tools and recorded all interviews. On average, the respondents had about six years of experience with LSAD, with a minimum of three years and a maximum of fifteen years. The experts' roles included security leads of agile programs, security engineers and security champions, an IT (security) consultant, an IT (security) architect, and a product owner (PO). To protect the anonymity of our interviewees, we intentionally do not provide further details.

Data Analysis. There are several standardized methods for the analysis of qualitative material. We used the Kuckartz [20] model to analyze our interview study data because it offers a deductive-inductive possibility for coding classification formation. We conducted the content-structuring qualitative content analysis using the qualitative data analysis software *MAXQDA* [26]. The two researchers who performed the interviews also conducted the analysis.

4 Interview Results

In this section, we present the main findings from the data analysis of the expert interviews. We first overview our results, then summarize framework usage and challenges, followed by the findings in the four categories of our interviews. To ensure the anonymity of the participating organizations, we intentionally describe the results only in an aggregated format and not specific for each case, except for Table 1.

4.1 Overview

Table 1 contains a summary of the results. We identified and selected recurring best practices that emerged from the interview analysis. We classify and visualize them according to their usage in each organization through *harvey balls*. The table does not represent a complete summary, but we filtered our results for two main cases. First, the concepts with the highest recurrence, and second, concepts with the highest ratio of conflicting viewpoints among the experts. We thus prioritize displaying the most important findings based on these two criteria.

4.2 Frameworks and Challenges

Scaled Agile Framework Usage. In the beginning, we asked about the scaled agile frameworks used in the organizations. Two experts stated that their organizations adhere to the guidelines of a specific framework, in one case LeSS [22], in the other case SAFe [43]. A third and fourth expert described a more heterogeneous agile landscape where teams choose frameworks individually depending

Table 1. Overview of recurring best practices

	01	02	03	04	05	06	07	08	09
Integration of security activities									
Security self-assesment	●	●	◑	○	◑	●	●	●	◕
Bug bounty		◑		◑	●	○	●	○	●
Threat modeling	◑	●	●	●	◕	◑	◑	●	◕
Penetration testing	●	●	●	●	●	●	●	●	●
Security audits		●	●	●		●	●	◑	●
Security code review	◑	◑	●	○	●	●	○	●	◑
Tool-support and automation									
DevSecOps pipeline	◕	●	●	◕	●	◑	◑	●	●
Static code analysis	◑	●	●	●	◑	○	◑	●	●
Vulnerability scanning	◑		●	◕	●	●	◕	●	◑
Dependency checks	●	◑	●	◕		○	◑	○	◑
Security governance									
Bottom-up	◑	○	○	○	◑	○	◑	◔	◔
Top-down	●	●	●	●	●	●	●	●	●
Reusable components		●	●		◑	○	◕	◕	◑
Organizational structure									
Security champion	◑	●	○	○	●	○	●	◕	◑
Security engineers or architects	◕	●	◕	●	◑	●	●	●	●
Central security teams	●	●	◕	◕	●	◑	●	○	◕
Communities of practice	◕	●		◑	◕	●	●	●	●

none: ○ | rare or planned: ◔ | partial: ◑ | frequent: ◕ | complete: ●
no classification possible: *empty*.

on the requirements. Two experts stated that no "textbook framework" is being used for scaled agility. The remaining three experts indicated that their organizations built their own frameworks, including parts of established frameworks.

Security Challenges in LSAD. Initially, we also asked the participants about the main challenges related to security in their LSAD environment. However, we will only present challenges mentioned by at least three independent experts. The first challenge is the lack of personnel with sufficient experience in both security (governance) and agile software development. The scaled agile environment amplifies the problem because centralized security teams have frequent contact with agile teams due to short development cycles. Also, the expected response times of security experts to inquiries of agile teams are lower, resulting

in a higher pressure on central security experts and possible frictions and delays in the development process.

The second challenge is the conflict between security governance and team autonomy when coordinating many teams. Teams should work as autonomously as possible, yet security policies and standards must be defined and managed. Scaling makes it challenging to monitor and control, as it is no longer possible to "look over the shoulders of the developers", as one expert stated.

4.3 Organizational Structure

All interviewed experts report that their organization is performing some sort of structural adaptations of their agile programs due to a higher relevance of security. Figure 1 shows a generalized summary of the results.

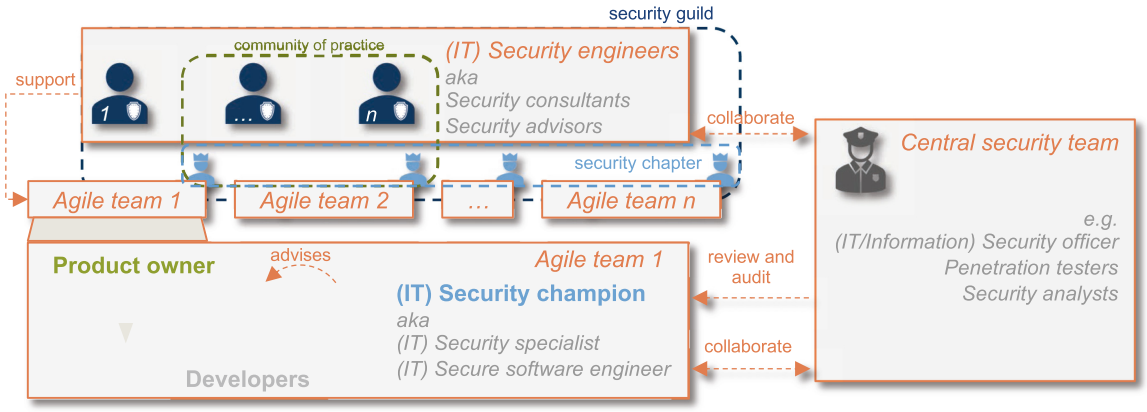

Fig. 1. Overview of organizational structure of agile programs

Centralized Security Teams. A common theme between the experts, with one exception, is that their organizations leverage existing central security teams to work with agile programs. These teams include individuals dedicated to security, e.g., penetration testers, security analysts, or information security officers. Centralized teams set overarching security quality criteria for deployments of software product increments and perform security verification. They also identify and handle compliance issues, perform risk analyses and security reviews (e.g., code review or penetration tests). Some activities such as threat modeling are performed collaboratively with individual development teams. This collaboration is beneficial for training purposes. The achieved knowledge transfer might enable agile teams to perform these activities by themselves in the future, reducing the workload of central teams. Depending on the criticality and security requirements of the software artifact, some of the analyzed organizations use central security teams for auditing and approving release-ready changes before deployments to production environments. Both threat modeling and reviews are discussed in more detail in Sect. 4.6. Members of central teams are often focused

on a product area or specialized in a specific security topic or technology. As mentioned in the challenges in Sect. 4.2, central teams face scaling issues and become a bottleneck when collaborating with agile development teams.

This bottleneck motivates the introduction of new roles within the agile programs. The goal is to reduce the workload on central teams and, more importantly, increase the security capabilities and thereby the autonomy of agile teams. Based on the collected data, we distinguish between two types of security-focused roles, *team-internal* and *team-external*.

Team-Internal Roles. These agile team members continue to be developers but receive additional security training. The analyzed cases use designations such as security champion, security specialist or secure software engineer, hereafter referred to only as security champion (SC). They provide the benefit of increasing security awareness. As developers, they know their products and are also familiar with security standards and best practices. One interviewee stressed that it is essential to clarify that the whole team is still responsible for the security of their application. The SC takes the lead on security activities, serves as a fixed contact person to communicate with team-external parties, and advises other team members and the PO. Three cases do not use an SC and rely more on other measures such as automated security testing.

Team-External Roles. They are referred to as security engineers, security consultants or security advisors, hereafter referred to only as security engineer (SE). They support two to twenty teams with security expertise and are often placed between the development teams and a central security department, acting as facilitators. In some organizations, SEs conduct threat modeling workshops with development teams. In other cases, this is the responsibility of the SC, to prevent bottlenecks. SEs may also analyze laws, policies, and security best practices and ensure knowledge transfer to development teams. They specialize in a software stack or are assigned to specific development teams. Two of the analyzed cases currently have no plans to introduce a specialized security role. A solution architect is responsible instead.

Cross-Team Collaboration. Security knowledge sharing takes place through regular meetings and training. Some organizations use the concept of communities of practices. Others unite the previously described roles in so-called guilds or chapters. A difference is in the scope, frequency, and target audience for which these exchanges occur. Moreover, organizations use corporate social networks and wikis to share and document security knowledge and search for experts. However, knowledge sharing remains a challenge. Existing documentation is not always helpful due to its complexity or lack of specific details for certain combinations of platforms and software. According to one expert, providing code examples for security topics is most helpful for developers.

4.4 Security Governance

All analyzed companies mainly rely on a top-down governance approach. In most cases, centralized security governance teams create company-wide stan-

dards from applicable regulations, international standards, and best practices. The companies differ in how development teams can participate in shaping security governance. One interviewee explicitly stresses that individual teams should not influence security governance because they should prioritize the development of their product. Others grant development teams a limited say in the governing standards, allowing a partly bottom-up approach. In those cases, agile teams support shaping internal standards adjustments with sufficient justification. A promising approach for effective security governance in LSAD is providing standardized, security-focused components that teams can reuse. Interviewees mentioned that these components also simplify application security verification. Stated examples are identity and access management, validation of inputs, encryption of data, or secure communication. Challenges include outdated documentation, uncertainties about correct usage, and lack of awareness.

4.5 Tool Support and Automation

All interviewees stated that their companies use DevSecOps pipelines for their applications' build and deployment phases.

Static Application Security Testing. A common denominator is the use of static code analysis tools, which are mandatory to varying degrees. In some companies, the usage depends on project requirements and the development team's decisions. In others, it is compulsory for all applications. Depending on the criticality of the findings, teams have to meet different thresholds to deploy changes to production. False positives are a commonly reported challenge of static security testing. They are especially problematic because they may lead to developers ignoring analysis results. A particular form of static analysis is using automated dependency checks, e.g., to look for the usage of outdated open-source libraries that could introduce new vulnerabilities into the product.

Dynamic Application Security Testing. The use of dynamic application security testing is not yet as mature as static code analysis. The experts stated that there are initiatives to evaluate and establish dynamic application security testing tools. They aim to automate parts of manual penetration tests. Furthermore, the experts mentioned the use of regular vulnerability scans, e.g., to check the infrastructure of the development teams for unnecessary open ports, insecure TLS versions or cipher suites, insecure HTTP header, or other security misconfigurations. Usually, central teams provide these scanning tools. Reports are immediately made available to development teams or at regular intervals, depending on the criticality.

Metrics and Quality Gates. Automation tools that are part of a DevSecOps pipeline provide metrics, e.g., for automated deployment decisions. Those metrics might include the number of open findings, the average criticality, or a total score. For these metrics, the experts stress the importance of agreeing on thresholds for quality gates. These thresholds set the boundary of whether an application is likely to be secure enough to release to production. Due to the limited capabilities

of automated tools, experts stressed not to rely exclusively on automation. As an outlook, one interviewee noted that the increasing use of machine learning might soon blur the line between the areas of security testing that can be automated and those that cannot.

4.6 Integration of Security Activities

Performing concrete activities to directly or indirectly increase the degree of security of a software product is crucial. The focus of the interviews was especially on which activities are most suitable in LSAD environments, and discussing their benefits and drawbacks. The following activities were the most discussed ones by our interviewed experts.

Code Reviews and Pair Programming. Most companies use code reviews as a form of manual intervention in developing secure applications. In two cases, pair programming is used instead as the primary quality assurance activity. A reported challenge in multiple analyzed cases is that code reviews usually deal with code quality in general (except for dedicated security code reviews), and security aspects may frequently fall short. One expert explained that they focus on automated static code analysis due to the high time consumption of code reviews. Also, other experts mentioned that code reviews are a trade-off between cost and the prospect of higher code quality. Nevertheless, one expert calls code reviews "the most pragmatic approach to developing secure software". The extent and frequency of code reviews vary. Some companies decide based on the criticality and required level of protection of the software product, while others leave it to the development teams. Especially when deploying critical code to production, organizations tend to mandate code reviews. Experts mentioned that it would be helpful to conduct security code reviews only if there was a security-relevant change. However, the crux lies in identifying those relevant changes, but automation may help in the future.

Penetration Tests and Bug Bounty Programs. All case companies regularly perform penetration tests. Both internal teams, as well as contractors, are used for this purpose. The frequency and scope vary depending on the product's criticality and size. The primary reported challenge of penetration testing is the lack of continuity because of the necessary preparation and follow-up work. Short penetration tests that only assess the changes of a smaller product increment are usually not seen as economically viable. Bug bounty programs are a valuable alternative to detect vulnerabilities continuously and provide the advantage of scaling through crowd-sourced security testers.

Security Reviews and Audits. Companies use security reviews to assess compliance with internal and external regulations. Depending on the criticality of the application, the audit frequency varies from quarterly to yearly. Reviews might include assessing system architecture or security documentation, code reviews, or penetration tests. A distinction can be made between pre-deployment and post-deployment audits. A hybrid approach is also possible, e.g., regularly using

post-deployment audits and applying pre-deployment controls every few sprints, or only if a product recently failed security audits. For low-risk applications, code can be deployed before all checks have been performed. When assessing the compliance of an application with given standards, respondents pointed to the commitment to guidelines. Some are merely recommendations, while others are considered indispensable.

Threat Modeling. Because of its good fit for iterative software development, threat modeling has a high priority for the interviewees. It can be performed during the initial design phase. For continuous integration into short sprints, delta threat modeling is performed. Delta threat modeling focuses on changes of the increment. The results of threat modeling can be used to prioritize specific components for code reviews or penetration testing.

Security Self-assessments. There are two main usages for security self-assessments. First, to determine whether the product in development is compliant with policies and guidelines. Second, to determine the security relevance and criticality. Self-assessments can be an efficient tool at scale because they delegate responsibility to the teams. One interviewee stressed that the goal is to keep the number of validations by team-external stakeholders as low as possible. A benefit of self-assessments is the creation of security awareness. The concept of "comply or explain" was also mentioned. Developers may explain where they have made a conscious decision not to meet a requirement. Depending on the criticality, this might be considered during risk management. One organization deliberately avoids self-assessments because they are too time-consuming.

Security Risk Management. A recurring aspect in the interviews is the possibility to release or keep operating software with certain security risks or compliance issues, often referred to as "risk acceptances". A PO has to take responsibility for the risk and systematically document it. A SC or SE usually supports the PO to identify and report risks proactively. Furthermore, risks can also result from other activities, e.g., threat modeling, penetration testing, or security reviews. Some teams perform and document risk assessments themselves, e.g., as attributes or flags of their feature tickets or user stories.

Security Documentation. On the on hand, experts stated that extensive security documentation is often not feasible for frequent product iterations. Therefore, companies evaluate tools to automatically create documentation, e.g., risk reports generated from threat models. On the other hand, experts explained that incrementally adapting and extending existing documentation with every sprint is feasible. They suggested using existing tools to include security requirements, e.g., issue tracking software.

5 Discussion

We answer our research question by discussing the key findings and then critically describe the limitations.

5.1 Key Findings

We identified two current challenges specific to security in LSAD that at least three experts mentioned. The first challenge is the lack of qualified personnel with sufficient experience in both security (governance) and agile software development. This challenge amplifies in LSAD due to the larger number of teams. The second challenge is the conflict between security governance and team autonomy when coordinating many teams.

An essential aspect addressing the first identified challenge is the structure of the agile program. Our findings show that all analyzed cases introduce additional security roles, as recommended in the literature. We were able to identify the use of central security teams, roles within the development team, and roles outside of a team. Furthermore, we show that some organizations are not leveraging team-internal security roles, such as a SC. Nevertheless, these roles might be most effective long-term because they enable teams to perform more security activities independently, resulting in more autonomy. To support agile teams, a solid DevSecOps pipeline with static and dynamic application security testing tools is indispensable.

The second challenge fits well with our findings in the security governance category. In all of the analyzed cases, security governance is mainly driven top-down, in contrast to the recommendations from the literature. However, bottom-up approaches are beginning to establish, e.g., development team members gathering in dedicated security communities. In our opinion, leaving the definition of security standards up to individual teams results in substantial, economically unjustifiable efforts and might result in conflicts of interest. A certain level of top-down control is still necessary, e.g., to prepare for external audits. Nevertheless, agile teams should be able to influence the security governance decision-making, and top-down governance should partly shift to self-governance. The described security roles provide a good starting point for building the necessary competency in and around agile teams. This shift could be a way to find the right balance between autonomy and control, consequently bringing closer security governance and LSAD.

Finally, we would like to place our results in the context of the security challenges described by Amber et al. [48], and existing software security maturity models. Our findings regarding the structure of the agile program, security governance, and security activities provide more clarity on how to address the challenge of aligning security objectives in a distributed setting, and contribute to solving the challenge of a common understanding of roles and responsibilities. Our results in the tool-support and automation category relate to the third challenge described by Amber et al., which is "the integration of low-overhead security testing tools" [48].

We identified common patterns between our results and established software security maturity models. For example, the BSIMM [28] identifies so-called *software security groups* in the studied organizations, which are described very similarly to the observed centralized security teams in our study. Another example is the *satellite* role, whose description is largely consistent with the team-internal

roles reported in our study. In this particular aspect, our study provides even more granularity by identifying and describing the team-external roles, which are even more widespread than the team-internal roles in the LSAD environments analyzed in this study. Further research on the similarities and differences between our results and software security maturity frameworks could lead to additional interesting findings.

5.2 Limitations

Even though we conducted an interview study, some of the common limitations of case studies described by Runeson and Höst [40] are also relevant for our study and help to structure our limitations. We addressed the threat of *construct validity* by clarifying any ambiguity directly during the conversation with the interviewees. To overcome the threat of *external validity*, which refers to a limited generalizability of results, we based our interviews on scientific literature and conducted the interviews in nine organizations from five industries. However, since we interviewed one expert at each company, we have only a limited picture of each organization. Companies are rarely homogeneous enough for one expert to grasp the entire situation. We countered this by designing our questions to identify overarching patterns within an organization. Additionally, we encouraged our interviewees to keep generalizability in mind. Moreover, the total number of interviewees might be considered relatively small. However, we had already reached a certain level of saturation in the sense that the data collected in the last few interviews became increasingly redundant compared to the data previously collected. To ensure *reliability*, we recorded, transcribed and coded the interviews. This analysis was documented, validated and discussed by the two researchers. Finally, typical problems arise when conducting interviews. That is why we followed the guidelines for good interviews by Kvale [21].

6 Conclusion and Future Work

Addressing security in LSAD is a significant challenge. Despite the importance, there is a paucity of research. Therefore, this paper provides insights into the research question of how security is addressed in LSAD by presenting the results of an interview study. We conducted a literature review to categorize the research topic and interview guide, resulting in four categories: agile program structure, security governance, security activities, and tool support and automation. Our interviews were conducted with nine experts from nine organizations in five industries. One of the key findings is that organizations use centralized security teams, team-internal and team-external security roles. In addition, organizations are using automation for security testing and integrating security activities such as threat modeling or code reviews. Security governance is mainly top-down, while our recommendation is to shift attention to bottom-up approaches. Our findings contribute to raising awareness of the areas to focus on when developing secure software at scale. Practitioners could leverage our results by discussing and applying the identified best practices in their organizations.

Our research could serve as the basis for further scientific investigation. The recurring best practices could be analyzed for their relative impact and effectiveness. Due to the complexity of the research topic, further research could also identify and explore other important aspects regarding security in LSAD, in addition to the four categories identified in our work. Moreover, as we suggest a shift toward more bottom-up security governance, a more in-depth study or evaluation of existing approaches could be conducted. For example, further research could focus on the impact of relevant secure software development maturity models to adapt security governance and compliance processes to agile at scale. More mature development teams may be more capable to self-govern their security posture, and their organizations may be able to afford less top-down control.

Funding. This work has been supported by the German Federal Ministry of Education and Research (BMBF) Software Campus grant 01IS17049.

References

1. Alsaqaf, W., Daneva, M., Wieringa, R.: Quality requirements in large-scale distributed agile projects – a systematic literature review. In: Grünbacher, P., Perini, A. (eds.) REFSQ 2017. LNCS, vol. 10153, pp. 219–234. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54045-0_17
2. Ambler, S.W.: Agile software development at scale. In: Meyer, B., Nawrocki, J.R., Walter, B. (eds.) CEE-SET 2007. LNCS, vol. 5082, pp. 1–12. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85279-7_1
3. Ambler, S.W., Lines, M.: Choose Your WoW. Boston (2020)
4. Aon PLC: 2019 global risk mangement survey - report — Aon (2019). https://www.aon.com/getmedia/8d5ad510-1ae5-4d2b-a3d0-e241181da882/2019-Aon-Global-Risk-Management-Survey-Report.aspx. Accessed 21 Feb 2021
5. Arkin, B., Stender, S., McGraw, G.: Software penetration testing. IEEE Secur. Priv. **3**, 84–87 (2005)
6. Barbosa, D.A., Sampaio, S.: Guide to the support for the enhancement of security measures in agile projects. In: 2015 6th Brazilian Workshop on Agile Methods (WBMA), pp. 25–31 (2015)
7. Beznosov, K., Kruchten, P.: Towards agile security assurance. In: Raskin, V. (ed.) Proceedings of the 2004 Workshop on New Security Paradigms, ACM Conferences, p. 47. ACM, New York (2004)
8. Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K., Kruchten, P.: Extending XP practices to support security requirements engineering. In: Bruschi, D. (ed.) Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems, ACM Conferences, p. 11. ACM, New York (2006)
9. Breaux, T.D., Anton, A.I.: Analyzing regulatory rules for privacy and security requirements. IEEE Trans. Software Eng. **34**, 5–20 (2008)
10. Dännart, S., Moyón, F., Beckers, K.: An assessment model for continuous security compliance in large scale agile environments: exploratory paper. In: Advanced Information Systems Engineering, pp. 529–544 (2019)
11. Digital.ai: 15th annual state of agile report (2021). https://digital.ai/resource-center/analyst-reports/state-of-agile-report. Accessed 21 Feb 2021

12. Dikert, K., Paasivaara, M., Lassenius, C.: Challenges and success factors for large-scale agile transformations: a systematic literature review. J. Syst. Softw. **119**, 87–108 (2016)
13. Döring, N., Bortz, J.: Forschungsmethoden und Evaluation in den Sozial- und Humanwissenschaften. S, Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-642-41089-5
14. E. U. A. for Cyber Security.: Enisa threat landscape 2020 (2020). https://www.enisa.europa.eu/topics/threat-risk-management/threats-and-trends. Accessed 21 Feb 2021
15. Google: Google security whitepaper - google cloud (2019). https://cloud.google.com/docs/security/overview/whitepaper. Accessed 21 Feb 2021
16. Horlach, B., Böhmann, T., Schirmer, I., Drews, P.: It governance in scaling agile frameworks. In: Tagungsband Multikonferenz Wirtschaftsinformatik 2018 (2018)
17. IT Governance Institute: Information Security Governance: Guidance for Boards of Directors and Executive Management. IT Pro, IT Governance Institute (2006)
18. Johnson, B., Song, Y., Murphy-Hill, E., Bowdidge, R.: Why don't software developers use static analysis tools to find bugs? In: Proceedings of the 2013 International Conference on Software Engineering, ICSE 2013, pp. 672–681. IEEE Press (2013)
19. Keramati, H., Mirian-Hosseinabadi, S.: Integrating software development security activities with agile methodologies. In: 2008 IEEE/ACS International Conference on Computer Systems and Applications, pp. 749–754 (2008)
20. Kuckartz, U.: Qualitative Inhaltsanalyse: Methoden, Praxis, Computerunterstützung. Grundlagentexte Methoden, Beltz Juventa, Weinheim and Basel, 4. auflage edn. (2018)
21. Kvale, S.: Doing Interviews, The Sage Qualitative Research Kit. Flick, U. (ed.), vol. Pt. 2. SAGE, Los Angeles (2007)
22. LeSS: Overview - large scale scrum (less) (2022). https://less.works/. Accessed 21 Feb 2021
23. Luna, A., Kruchten, P., E. Pedrosa, M.L.D., Almeida Neto, H.R., Moura, H.P.M.: State of the art of agile governance: a systematic review. Int. J. Comput. Sci. Inf. Technol. **6**, 121–141 (2014)
24. Luna, A., Kruchten, P., Riccio, E., Moura, H.: Foundations for an agile governance manifesto: a bridge for business agility. In: 13th International Conference on Management of Technology and Information Systems, São Paulo, SP, Brazil (2016)
25. Ma, Z., Schmittner, C.: Threat modeling for automotive security analysis. Security Technology 2016, pp. 333–339 (2016)
26. MAXQDA: Maxqda — all-in-one qualitative & mixed methods data analysis tool (2022). https://www.maxqda.com/. Accessed 21 Feb 2021
27. Microsoft IT: Security for modern engineering: information security & risk management (2016). https://www.microsoft.com/en-us/download/details.aspx?id=54092. Accessed 21 Feb 2021
28. Migues, S., Erlikhman, E., Ewers, J., Nassery, K.: Building Security in Maturity Model (BSIMM) Report - Version 12 (2021). https://www.bsimm.com/
29. Mohan, V., Othmane, L.B.: SecDevOps: is it a marketing buzzword? - Mapping research on security in DevOps. In: 2016 11th International Conference on Availability, Reliability and Security (ARES), pp. 542–547. IEEE (2016)
30. Moyón, F., Méndez Fernández, D., Beckers, K., Klepper, S.: How to integrate security compliance requirements with agile software engineering at scale? In: Product-Focused Software Process Improvement, pp. 69–87 (2020)

31. Moyón, F., Almeida, P., Riofrío, D., Méndez Fernández, D., Kalinowski, M.: Security compliance in agile software development: a systematic mapping study. In: 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 413–420 (2020)
32. Myrbakken, H., Colomo-Palacios, R.: DevSecOps: a multivocal literature review. In: Mas, A., Mesquida, A., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2017. CCIS, vol. 770, pp. 17–29. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67383-7_2
33. Newton, N., Anslow, C., Drechsler, A.: Information security in agile software development projects: a critical success factor perspective. In: Proceedings of the 27th European Conference on Information Systems (ECIS) (2019)
34. Nguyen Quang Do, L., Wright, J., Karim, A.: Why do software developers use static analysis tools? A user-centered study of developer needs and motivations. IEEE Trans. Softw. Eng. **48**, 835–847 (2020)
35. OWASP Foundation: OWASP Software Assurance Maturity Model - Version 2.0 (2020). https://owaspsamm.org/model/
36. Oyetoyan, T.D., Cruzes, D.S., Jaatun, M.G.: An empirical study on the relationship between software security skills, usage and training needs in agile settings. In: 2016 11th International Conference on Availability, Reliability and Security (ARES), pp. 548–555 (2016)
37. Petersen, K., Wohlin, C.: The effect of moving from a plan-driven to an incremental software development approach with agile practices. Empir. Softw. Eng. **15**, 654–693 (2010)
38. Poller, A., Kocksch, L., Türpe, S., Epp, F.A., Kinder-Kurlanda, K.: Can security become a routine? In: Lee, C.P. (ed.) Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, pp. 2489–2503, ACM Digital Library. ACM, New York (2017)
39. Rindell, K., Ruohonen, J., Hyrynsalmi, S.: Surveying secure software development practices in Finland. In: Proceedings of the 13th International Conference on Availability, Reliability and Security, ACM Other Conferences, pp. 1–7. ACM, New York (2018)
40. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. **14**, 131–164 (2009)
41. Sadowski, C., Söderberg, E., Church, L., Sipko, M., Bacchelli, A.: Modern code review. In: Paulisch, F., Bosch, J. (eds.) Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice, pp. 181–190. ACM, New York (2018)
42. SAP: The secure software development lifecycle at sap (2020). https://www.sap.com/documents/2016/03/a248a699-627c-0010-82c7-eda71af511fa.html. Accessed 21 Feb 2021
43. Scaled Agile Framework: Safe 5.0 framework (2022). https://www.scaledagileframework.com/. Accessed 21 Feb 2021
44. Shostack, A.: Threat Modeling: Designing for Security. Wiley, Indianapolis (2014)
45. Bartsch, S.: Practitioners' perspectives on security in agile development. In: 2011 Sixth International Conference on Availability, Reliability and Security (ARES), pp. 479–484 (2011)
46. Steghöfer, J.-P., Knauss, E., Horkoff, J., Wohlrab, R.: Challenges of scaled agile for safety-critical systems. In: Franch, X., Männistö, T., Martínez-Fernández, S. (eds.) PROFES 2019. LNCS, vol. 11915, pp. 350–366. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-35333-9_26

47. Sulejman, V.: It governance and its agile dimensions: exploratory research in the banking sector. In: Proceedings of the 52nd Hawaii International Conference on System Sciences 2019 (2018)
48. van der Heijden, A., Broasca, C., Serebrenik, A.: An empirical perspective on security challenges in large-scale agile software development. In: Oivo, M. (ed.) Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 1–4. ACM, New York (2018)
49. Vejseli, S., Rossmann, A., Connolly, T.: Agility matters! Agile mechanisms in it governance and their impact on firm performance. In: Bui, T. (ed.) Proceedings of the 53rd Hawaii International Conference on System Sciences. Hawaii International Conference on System Sciences (2020)

The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study

1st Sascha Nägele
Department of Computer Science
Technical University of Munich
Munich, Germany
sascha.naegele@tum.de

2nd Nathalie Schenk
Department of Computer Science
Technical University of Munich
Munich, Germany
nathalie.schenk@tum.de

3rd Florian Matthes
Department of Computer Science
Technical University of Munich
Munich, Germany
matthes@tum.de

***Abstract*—Agile methodologies have gained popularity in software and information systems engineering due to their ability to enable rapid adaption to changing requirements and ensure business value creation in fast-paced environments. However, scaling agile to multiple teams presents challenges related to security governance and compliance. Traditional security activities struggle to keep pace with iterative agile methods. The tension between security and agility intensifies in scaled environments as governance and compliance procedures conflict with the desired autonomy of agile teams. With the increase in the number and complexity of security risks, it is imperative to better understand the current challenges and solution approaches for security governance in large-scale agile development (LSAD). To this end, we conducted a systematic literature review and an interview study involving nine industry experts. We identified 15 relevant challenges and analyzed existing LSAD frameworks concerning their solution approaches for achieving security governance and compliance. In addition, we contribute an overview of alternative solution approaches and propose five factors to balance control and autonomy to mitigate security challenges in LSAD. Our findings provide a foundation for developing well-grounded solution artifacts that address the identified challenges.**

***Index Terms*—large-scale agile, software development, security, systematic literature review, interview study**

I. Introduction

Organizations must adapt to uncertain, demanding, and rapidly changing requirements to ensure business value when engineering software and information systems. Agile software development methods facilitate high-quality software development in such environments [1], [2]. Large-scale agile development (LSAD) methods have been increasingly adopted to leverage these benefits in larger projects and organizations [3]–[5]. However, scaling agile is complex and demanding [6], [7], requiring additional coordination mechanisms and systematic integration of quality assurance [7], [8].

At the same time, increasing security risks are among the most significant business risks to organizations [9], [10], with software engineering being a primary source of security vulnerabilities [11], [12]. The tension between security and agile methods is evident, with traditional security activities being often too slow for agile delivery speed [13]. Moreover, the increased development pace of agile methods may result in less secure products [14], [15]. The tension between security and agility intensifies in scaled environments as governance and compliance procedures collide with the desired autonomy of agile teams [16]. This tension is further reinforced by increasing regulatory requirements impacting software development and delivery processes [17]. Hence, the urgency to analyze the tension between LSAD, security governance and compliance is high. The literature emphasizes that LSAD frameworks such as the Scaled Agile Framework (SAFe) [18], Disciplined Agile (DA) [19], or Scrum@Scale [20] provide too little guidance on security governance and compliance [12], [21], [22].

This leads to an interesting gap in current research and practice. To our knowledge, there is no study yet that systematically analyzes the presented research area in both scientific literature and practice. We aim to answer the following two research questions:

- RQ1: What are the current challenges of security governance and compliance integration in LSAD?
- RQ2: What are existing solution approaches in literature and practice to integrate security governance and compliance in LSAD?

To provide an answer to these research objectives, we conducted a systematic literature review (SLR) and an interview study with nine experts. In addition, we performed a keyword-based LSAD framework analysis.

II. Foundations

IT governance is a crucial component of corporate governance and ensures that IT strategy and enterprise strategy are aligned and mutually beneficial. IT governance also includes guaranteeing that the usage of IT resources and the management of IT risks are taken care of responsibly [23], [24]. The growing importance of security results in "the need

This work has been supported by the German Federal Ministry of Education and Research (BMBF) Software Campus grant 01IS17049.

979-8-3503-1515-8/23/$31.00 ©2023 IEEE

for effective information security governance" [23]. Julisch defines *security compliance* as "the state of conformance with [...] imposed functional security requirements and of providing evidence (assurance) thereof" [25]. The sources of those requirements may be external (e.g., defined by the government, regulations, frameworks, or customers) or internal. With the term *security*, we refer to *information security*, which pursues the goal of protecting information against incorrect and inappropriate disclosure (confidentiality), alterations (integrity), or loss (availability) by mitigating risks to an acceptable level [26], [27]. Dikert et al. have reviewed different definitions of *LSAD* and, based on those findings, define a LSAD environment as a "software organization with 50 or more people or at least six teams" [8].

III. Research Methodology

We combined a systematic literature review (based on [28], [29], [30]) and an interview study (based on [31], [32], [33]) for systematic data collection to gain insights from literature and practice.

A. Systematic literature review

Our goal was to identify and analyze the current state of security compliance and governance in LSAD. We aimed to categorize, combine, and synthesize related findings to gain new knowledge and identify related work. We extracted the main steps of the procedures proposed by Webster and Watson [28], Brocke et al. [29], and Kitchenham and Charters [30]. Figure 1 shows an overview of our SLR approach. Initially, we

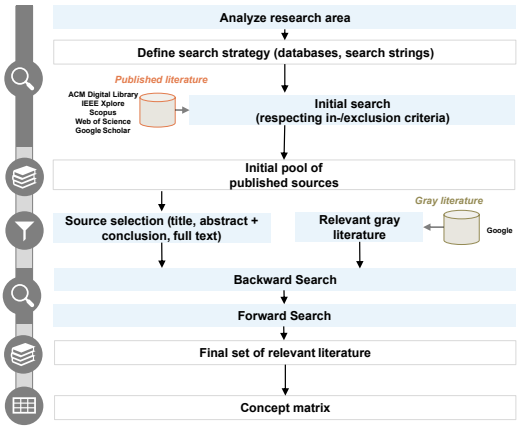

Fig. 1. SLR approach

investigated the research area by scanning essential journals and analyzing the references of known and relevant articles [28], [30]. We then defined and applied a systematic search strategy, including the selection of databases and formulation of search strings [29], [30].

Firstly, to capture the literature comprehensively, we identified various electronic sources to query [29], [34]: *ACM Digital Library*, *IEEE Xplore*, *Scopus*, *Web of Science*, and *Google Scholar*. These sources were deemed relevant within the software engineering domain, with the first two being more topic-specific than the others [35]. Using different databases ensured a more comprehensive literature search [36]. Secondly, we determined suitable keywords and search strings by conducting trial searches, considering synonyms or different spellings. We combined various sub-strings using boolean AND and OR connectors [30], [36], leading to the following search string: *agile AND (scale* OR "large-scale" OR scaling) AND (secur* OR DevSecOps OR SecDevOps OR "secure DevOps" OR safety)*.

Additionally, we used the following string for a complementary search: *agile AND (scale* OR "large-scale" OR scaling) AND (governance OR complian* OR assurance OR audit)*. We included this separate second search term because a combined search term was too long for some databases and also led to many irrelevant results. To limit the search results, we applied inclusion and exclusion criteria [30] as shown in Table I.

TABLE I
SLR inclusion and exclusion criteria

Inclusion criteria
• Published after 2001 • Matching search string within title, abstract, or keywords • Full-text available online and accessible • Written in English • Published in journals, conferences, or workshops • Published as (research) articles, (short, conference) papers
Exclusion criteria
• Duplicates • Insufficient relevance to answer research questions • *Google Scholar only: Not within first 100 hits* • *Scopus only: Not within subject area computer science*

We chose to include publications starting from 2001 when the *Agile Manifesto* [37] was published. To obtain the most relevant search results, we decided to search only within the title, abstract, or keywords of sources [36]. We selected sources by (i) reviewing the title for relevance, (ii) reviewing the abstract and conclusion for relevance, and (iii) if not rejected until then, reviewing the full text. To ensure completeness of the knowledge base, we performed a backward search to identify missing important literature by reviewing citations of the identified sources [28], [29]. A forward search was subsequently conducted, analyzing sources that cite relevant literature [28], [36]. We additionally included gray literature, such as white papers, blog articles, or market research reports, which we identified during the backward and forward search and the initial screening of the research area on Google [30], [35]. Gray literature ensures consideration of "the state of the practice" [38], which is especially important in practitioner-oriented research areas [38] like security in LSAD. The fact that there is not yet a large body of academic literature also argues for including gray literature [35]. Once neither the search in additional databases nor the forward and backward search yielded further relevant concepts, we assumed that the body of literature was complete [28]. The same applied to the search for gray literature [35]. Finally, the literature found was subject to analysis and synthesis [29]. Table II shows

the amount of relevant literature for each stage and database. We documented our SLR results in two concept matrices, one for each research question. The different concepts are grouped thematically and can consequently be combined and summarized [28].

TABLE II
OVERVIEW OF LITERATURE SEARCH RESULTS

	Search String 1		Search String 2	
Resource	**Hits**	**Relevant**	**Hits**	**Relevant**
ACM Digital Library	20	2	23	3
Google Scholar	100	7	100	12
IEEE Xplore	48	4	36	6
Scopus	131	11	102	15
Web of Science	98	8	71	12
After duplicates (per String)		12		19
After duplicates			24	
Backward search			22	
Forward search			4	
Added literature			9	
Total			**59**	

B. Interview study

Interviewing experts with experience in the problem of interest offers valuable insights that complement the knowledge gained through a literature review [32]. We conducted nine semi-structured interviews with experts from seven companies, focusing on LSAD environments concerning security or governance. Our interview study procedure primarily relies on Myers and Newman [31], H.J. Rubin and I.S. Rubin [32], and Saldaña [33]. Figure 2 shows an overview of our interview study approach. Semi-structured interviews allowed

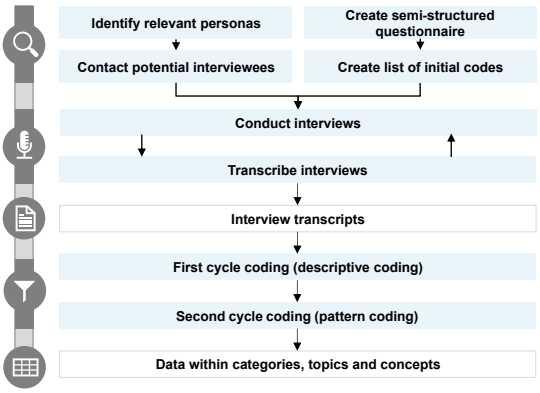

Fig. 2. Applied interview study approach

for follow-up questions, seeking clarifications or examples [31], [32]. The selected interviewees have experience in LSAD environments with a particular focus on security or governance. We sought consent to record and transcribe interviews and employed a cyclic coding approach as recommended by Saldaña [33]. Initial coding occurred in the first cycle, while the second cycle refined data by focusing on relevant aspects, enabling the development of accurate, conceptual codes [33]. We used descriptive coding in the first cycle and pattern coding in the second [33]. Deductive and inductive coding were combined, with initial codes derived from research objectives and questionnaires, while additional codes were identified during the coding process [39]. To facilitate coding, we utilized *MAXQDA* for efficient data processing and storage [33]. We merged the SLR and interview results into two combined concept matrices, allowing for a comprehensive understanding of the research area.

IV. SECURITY CHALLENGES IN LSAD

In our research aiming for a comprehensive understanding of security challenges in LSAD (*RQ1*), we identified three types of relevant challenges:

1) Security challenges in LSAD
2) Security challenges in (small-scale) agile development, transferable to LSAD
3) General challenges in LSAD, transferable to a security context

Since the first type is scarcely found in the literature (with only two relevant publications identified, namely [14], [40]), we included the second and third types in our analysis. We assessed the relevance of these challenges for security (governance and compliance) in LSAD based on literature findings, expert interviews (cited as I1-I9), and our research and industry experience. Figure 3 shows an overview of the identified challenges.

A. Challenges unique to security in LSAD (C1-C6)

Van der Heijden et al. [14] identify unique security challenges in LSAD, such as aligning security objectives in distributed settings, fostering a shared understanding of roles and responsibilities, and integrating low-overhead security testing tools. In addition, defining security requirements is complex, with prioritization, assignment to increments, and testing being difficult due to increased team and product dependencies. Activities like threat analysis or issue management are also challenging in LSAD [40]. Executing security practices at scale is generally more demanding, and the scaled context exacerbates the shortage of security practitioners, increasing the need for security automation [40]. However, the automation setup itself also requires experienced security practitioners. Security compliance in LSAD is a complex endeavor, with organizations often lacking a shared understanding of security governance and compliance [40]. The absence of agreed definitions for key concepts, such as the Definition of Done (DoD), hinders the integration and governance of security requirements at scale. Our interview study findings align with these results, indicating that a lack of well-defined processes, roles, and resources for security governance and compliance often inhibits scalability in LSAD.

B. Relevance of small-scale agile security challenges

In the following, we describe the results of our analysis of small-scale agile security challenges and their relevance in LSAD contexts.

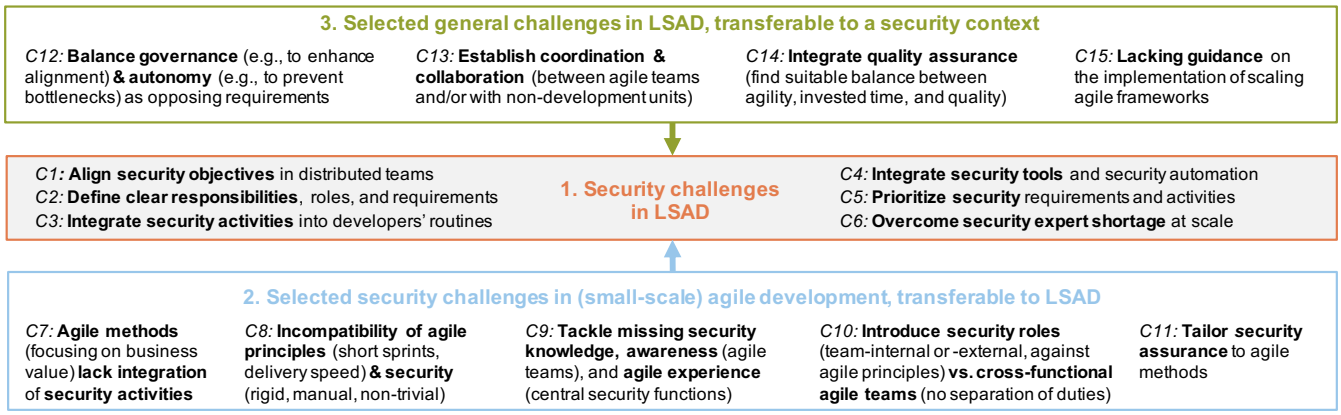

Fig. 3. Overview of identified security challenges in LSAD

1) Agile methods do not include security activities (C7): Agile methods primarily focus on fulfilling functional requirements and delivering business value [41], often neglecting security, an implicit quality issue that is not a direct business goal [11]. Consequently, security activities, such as penetration testing, threat modeling, or secure code reviews, are often overlooked or addressed minimally [14], [42]–[44]. Although integrating security reduces long-term development costs, justifying its inclusion in agile processes is difficult due to increased initial costs and the opaque return on investment [12], [43], (I3, I7). Management may perceive security as a resource-consuming aspect that detracts from feature development [11]. Hence, the challenge lies in cost-efficiently integrating security and agility [43]. Our interview findings reveal that security is frequently neglected or addressed post-development, reinforcing the notion that "security delays and slows down the process" (I3) and stressing its importance at scale.

2) Incompatibility of agile principles and security activities (C8): Traditional security activities often struggle to keep pace with agile methods' rapid delivery [13]. The iterative nature of agile methods poses challenges for applying security activities, such as risk analysis [43] or external stakeholder reviews [44], which may create bottlenecks due to their roots in linear development processes [13], [45]. Adapting security practices to cope with short iterations and frequent changes is often challenging [44]. Our interview analysis highlights the importance of (i) building security in from the beginning to reduce pre-deployment efforts, and (ii) employing automation and tool support. Interviewees emphasize that organizations should maintain software development flow by utilizing tool-supported security checks and minimizing hand-overs and manual review activities (I1, I3, I6). As LSAD extends agile methodologies to larger environments, this challenge applies to both small-scale agile and LSAD.

3) Missing security knowledge, awareness, and mindset (C9): Security is a knowledge-intensive area, and agile development's reliance on tacit knowledge [46] can pose problems. Product owners and developers may be unaware of security implications [14], [44], and some view security as a burden, and they may lack intrinsic motivation to address it [11], (I2). Central security teams can inadvertently contribute to security neglect in agile projects due to developers' previous negative experiences, such as delivery delays due to interventions of a central team [13], (I1). Our interview findings confirm that while development teams should be autonomous and capable of assessing risks and security implications, they often lack the necessary knowledge (I4). It is crucial to encourage agile teams to seek expert help when needed (I2, I3, I4) but central security teams alone cannot sufficiently guide multiple agile teams due to capacity constraints (I4, I7). Organizations should foster rudimentary security awareness (I3, I6), which is crucial yet challenging [14]. Some argue that higher awareness leads to greater acceptance and understanding of security tasks (I3), while others emphasize capacity issues rather than awareness as the primary obstacle (I1). In LSAD, knowledge issues can impact multiple teams, magnifying their significance. Van der Heijden et al. [14] investigated these challenges in LSAD environments and classified them as relevant, but not unique to scaled agile, supporting our assessment.

4) Security roles in cross-functional agile teams (C10): In non-agile projects, team-external security experts typically handle security issues [43], [46]. However, in agile environments with short iterations and continuous changes, integrating team-external experts may be undesirable [43], potentially causing bottlenecks [13]. Goertzel et al. [44] suggest a team-internal security expert, but agile approaches do not anticipate specialized roles, making integration challenging. Moreover, security contexts, particularly in regulated areas, often require role separation, which is not inherent in agile environments (I2). Organizations face challenges regarding accountability for security actions without a dedicated security role [14]. These challenges have been reported as relevant in LSAD but are not considered unique to scaled environments [14].

5) Integration of security assurance to ensure compliance (C11): Integrating specific security assurance activities is challenging due to traditional methods relying on a lengthy list of tasks [45] that do not align with the agile life-cycle [42].

They are typically performed by specialists after development [47], but collaboration with third parties may be burdensome, as they must be consulted during every cycle [43], [47]. Frequent changes necessitate repeating assurance activities, which increases costs [44], [47]. Short iterations lead to a focus on measurable progress, complicating security assurance integration [42]. Low-overhead security documentation implementation poses another challenge [14]. While documentation is crucial for compliance, it conflicts with agile methods [47]. If teams are responsible for security measures but do not prioritize documentation, there may be negative compliance implications and insufficient knowledge sharing between agile teams and security personnel [14]. However, Beznosov and Kruchten [47] suggest some assurance practices are suitable for agile methods, such as internal code reviews, secure design principles, and standards. Partly automating testing activities (e.g., penetration testing) can address challenges with integration in short iterations. Beznosov and Kruchten also propose to introduce new agile-friendly security assurance methods or to conduct assurance only at specific iterations, particularly at the beginning and end of product development. Goertzel et al. [44] even argue that independent verification and validation of security is not feasible in agile contexts. These issues reflect general quality assurance challenges in LSAD projects [7], [8], [48], [49], suggesting security assurance challenges also occur in the LSAD context, as confirmed in the interviews.

C. Relevance of general LSAD challenges

Dikert et al. [8] and Kalenda et al. [7] investigated challenges and success factors in LSAD. To ensure relevance, we focused on challenges related to security, safety, DevOps, automation, governance, compliance, assurance, and audits, corresponding to our literature review search terms (see Section III-A). To expand our literature base, we considered challenges in scaling frameworks but limited our analysis to framework-agnostic challenges for generalizability. These challenges arise from the original design of agile approaches for small, co-located teams now being applied in broader contexts [5].

1) Governance and team autonomy (C12): Governance challenges arise in LSAD due to the coexistence of agile and traditional processes [8]. Existing deployment procedures and mandatory activities may be tedious for agile teams [50], e.g, documentation requirements or quality gates [8]. Approval procedures might even reduce security awareness since teams do not feel responsible in the case of problems occurring after approval and release. In addition, if teams are only allowed to self-govern releases with low-impact and low-risk changes, developers might misclassify changes as low-risk to bypass approval procedures, potentially leading to worse results than with more self-responsible approaches (I3, I7). A burdensome approval process might also lead to teams releasing less often and bundling several changes into one release [50]. Nevertheless, LSAD requires robust governance to ensure consistency in both development and deployment of software [51], which may not align with the expectations of companies introducing agile methods [49]. A complete transition from traditional to agile methodologies remains under debate [8]. Some interviewed experts suggest implementing a central governance for LSAD, similar to recommendations in the literature [52]. Nevertheless, they stress the need for adapting governance processes to the agile context. For example, manual quality gates pose an immense overhead on iterative agile methods and cause bottlenecks (I1, I2, I3, I8). However, regulated industries might require certain activities to ensure compliance, even if that slows down iterative development (I1, I2, I4, I8).

On the other hand, Damm [51] highlights the potential for team self-governance, ensuring alignment without centralized functions. Naidoo and Rikhsoto emphasize the importance of agile and compliance teams working effectively within sound governance structures, balancing autonomy and control [52]. Supporting roles, such as security testing experts, are crucial for collaboration [52]. Petit and Marnewick suggest measuring team autonomy to influence the release process [50]. According to the interviewees, organizations must define governance integration without hindering progress (I3) and determine which quality gates are really necessary for every sprint (I2). Also, one expert advocates decentralization to achieve more autonomy, but also the introduction of random sampling tests instead of quality gates (I2) to ensure compliance. Agile practitioners face challenges in convincing governance leaders to align compliance practices with new development methods, such as automated tests in build and deployment pipelines, particularly when they lack agile experience (I1, I2, I4, I5, I8). These challenges are also highly relevant specifically to security.

2) Coordination and collaboration (C13): The need for collaboration and coordination between agile teams intensifies at scale. Hence, it is challenging to find the right amount of governance and control, support, and autonomy [53]. Interfaces with non-development functions may hinder agile benefits if they resist adopting agile methods or fail to at least align with agile teams [8]. For example, as the delivery speed increases with iterative development, more frequent interactions with non-development functions may delay review and release activities [8]. Closely related is the question of how agile teams can be embedded into established, non-agile business processes [7], such as bureaucratic audit and compliance procedures [52], or how these internal processes can adapt to agile development methods [48].

Within our research area, investigating the interfaces to central non-development units is crucial, as security is one as well. Another coordination and collaboration example is the need for addressing the division of security responsibilities among teams working on the same product. Several experts cite challenges arising from coordination and collaboration (I2, I3, I8). Therefore, we also consider this challenge to be pertinent to the security context.

3) Quality assurance (C14): Decreased quality and accumulating technical debt may occur when scaling agile, making quality assurance vital for success [7], [8]. However, agile methods also offer benefits, such as more frequent

and earlier testing, due to smaller increments and frequent deliveries. Automated testing is essential for increased speed, but implementation is often limited [48], [49]. Developing adequate test specifications is complex, particularly when team members lack relevant knowledge [54]. In regulated sectors, organizations must balance reliability and frequent releases [48]. Moyón et al. stress that requirements must be met to achieve compliance with regulatory standards "while not limiting the speed and flexibility agile development methodologies promise" [40]. An example in this area is defining an appropriate DoD without creating lengthy checklists that slow delivery [54]. Addressing these issues is crucial from a security perspective, as security is a key quality attribute [11]. The interviewees also emphasized the need for proper DoD specification and automated security testing as part of quality assurance (I3, I7, I8).

4) Implementation of large-scale agile in practice (C15): Introducing agility at scale is challenging for organizations due to limited practical guidance from theoretical frameworks and literature [8], [49]. Customizing frameworks and adopting new roles can be difficult [48], [49]. For successful LSAD implementation, Dikert et al. [8] emphasize the importance of selecting a suitable approach, customizing it effectively, and providing employee training and coaching on agile methods. This challenge is also relevant from a security perspective, as our interviewees reported the need for concrete guidance to adapt frameworks to include security aspects, integrate security activities at scale, and adopt security roles.

V. (Security) Governance and Compliance in Scaling Agile Frameworks

Our literature review and interviews indicate that existing scaling frameworks lack sufficient guidance on security compliance and governance [12] and their implementation for achieving these goals [8], [49].

To evaluate these claims and partly answer *RQ2*, we assessed how LSAD frameworks address the identified challenges. We systematically examined their primary resources for insights into balancing governance, compliance, and agility, focusing on security as the primary use case. We searched for content related to *"governance"*, *"compliance"*, or *"secur*"*.

Table III categorizes our findings by framework and subject, presenting results according to the relevance of coverage, ranging from *"nothing (relevant) mentioned" (-)* to *"low"*, *"medium"*, and *"high"*. We classified coverage as "low" if a framework mentions a topic but only partially addresses our research area. "Medium" coverage refers to broader discussion with precise information on pertinent aspects, while "high" coverage entails concrete guidance for organizations on how to put concepts into practice.

Our analysis revealed that minimalist frameworks, such as LeSS [56], Nexus [55], and the Spotify Model [57], did not cover the searched topics at the time of our analysis. Scrum@Scale [20] provides minimal insight into our specific issues. In contrast, SAFe [18] and DA [58], both more extensive in scope, addressed all search terms with medium coverage. Nonetheless, only three of the six frameworks offered guidance on security-related topics, and none provided detailed implementation information.

TABLE III
Framework coverage analysis results

	"secur*"	**"compliance"**	**"governance"**
Scrum@Scale [20]	-	low	-
Nexus [55]	-	-	-
LeSS [56]	-	-	-
Spotify model [57]	-	-	-
SAFe [18]	medium	medium	medium
DA [58]	medium	medium	medium

In the following, we examine the insights provided by the frameworks in detail and discuss their relevance to the identified challenges and solution approaches.

1) Scrum@Scale: Scrum@Scale [20] does not explicitly address security or governance. It mentions the need for an independent compliance department, addressing the challenge of coordinating and collaborating with non-development functions. However, due to the insufficient guidance on collaboration between Scrum teams and the compliance department and the lack of consideration of other challenges, we consider the level of coverage to be "low".

2) SAFe: SAFe introduces the concept of business agility, incorporating functions such as security and compliance [18]. Built-in quality is a core value of SAFe, advocating for work acceptance only if quality concerns, including security and compliance, are addressed in regular workflows. SAFe emphasizes automation and suggests including security requirements in the product backlog as non-functional requirements. It assigns so-called system teams to establish infrastructure for automated security testing and suggests shared services in areas like data security for multiple agile release trains. SAFe encourages lean governance, implementing lean quality management systems for continuous compliance. Validation is integrated into incremental development activities, and only final validation activities and sign-offs are performed prior to release. In summary, SAFe addresses security, governance, and compliance dimensions, showing its comprehensiveness. However, it lacks concrete guidance on several topics, e.g., how to collaborate with the shared services and prioritize non-functional requirements. Thus, we assess the coverage as "medium".

3) DA: DA [58] stresses the importance of security and risk management, advocating for security engineers to assess risks, formulate security requirements, and conduct testing. DA suggests staffing specialists on individual teams and subject matter experts supporting multiple teams. It also proposes treating security as a non-functional requirement and emphasizes the significance of sufficient testing. It recommends using security testing tools (e.g., static and dynamic application security testing) and implementing threat modeling. Metrics from such tools can be an indicator of quality and can be used for

security ratings. Security guidelines are provided for standardizing secure product development, with regular updates and expert support. DA defines governance as an enabling function instead of "command and control" [58]. According to DA, "governance should push as much skill, knowledge, responsibility, and automation into delivery teams as they can" [58]. To support compliance, DA suggests using templates, static and dynamic analysis, pair programming, DoD, and informal reviews, with formal reviews when necessary for regulatory compliance. To summarize, DA addresses several challenges identified in our SLR and interview study. However, it lacks clear guidance on adopting these procedures, aligning security activities with agile methods, or distributing responsibilities between security experts and agile teams. Therefore, we rate the coverage for all three areas as "medium".

VI. Security in LSAD integration approaches

This section presents how current literature addresses the existing issues and highlights the areas most relevant to our research, thereby further answering *RQ2*. Our SLR yielded three relevant categories of integration approaches:

1) LSAD approaches focusing on security or lean governance
2) Small-scale agile approaches focusing on security
3) Secure software engineering practices adapted to agile methods

A. Enhanced LSAD approaches

The work by Moyon et al. [40], [59] is unique as they explicitly address the tension between security compliance and LSAD. They introduce *S²C-SAFe*, enhancing SAFe by adding roles, activities, and artifacts related to security requirements, secure implementation, and security verification and validation testing. Notably, it involves a product security level assessment and threat model creation. The model also includes security requirements in the backlog, corresponding to attack vectors from the threat modeling, with security experts supporting their analysis. They also propose using secure coding standards and their inclusion in the DoD and emphasize the importance of security training for product owners. The study results show that visualized models are helpful in the communication and collaboration of agile developers and security experts [40].

Poth et al. [60] propose an approach for software development in regulated environments that is also scalable to multiple agile teams. The main goal is to ensure compliance while "offering as much autonomy to agile teams as possible" and not hindering the delivery of products [60]. According to a product's identified and prioritized quality risks, the team must undertake defined activities during development to ensure that the product is of high quality and compliant. The timing and scope of quality checks depend on parameters such as team maturity. Results show that granting freedom to teams may increase their acceptance of introduced approaches. Petit and Marnewick [50] introduce a so-called *"Earn Your Wings"* approach that determines an agile team's autonomy and accordingly assigns accountability. The approach is inspired by a "pilot's ability to fly an aircraft using five levels" [50]. Team autonomy is assessed by various aspects, such as maturity of engineering practices, severity of previous incidents, and application criticality. The authors claim that the process accelerates the release process, and agile teams spend less effort on finding workarounds due to their gained accountability.

B. Enhanced small-scale agile approaches

While these approaches do not provide specific guidance at scale, they can provide valuable inferences and ideas that could be applied to LSAD contexts. Boldt et al. [61] are among the few researchers who study how to integrate security into an agile software development process in the industry. They conclude that organizations must systematically address security issues during development to mitigate risks and integrate additional security roles, activities, and guidelines. Evaluation results show a positive risk management impact while it "at the same time maintains the development project's security cost at a reasonable level" [61]. Hanssen et al. [62] introduce *SafeScrum* and Fitzgerald [63] et al. *R-Scrum*. While both address safety-critical environments, the findings are still relevant as security is crucial in safety-critical development.

C. Adapted secure software engineering practices

To complete the overview of solution approaches, we include adapted secure software engineering practices. One of the most famous set of such practices is the *Microsoft Security Development Lifecycle* [64]. In the adapted variant for agile development [65], teams must not fulfill every requirement within every sprint or release. Instead, activities are divided into buckets, which are then distributed over time. Another important resource is *SAMM Agile* [66], published by the OWASP Foundation. It explains how organizations can integrate security activities into sprints, how collaboration between security teams and developers is achieved, and how to leverage automated testing. It also proposes introducing a security champion and stresses that teams should seek autonomy.

VII. Security responsibility influencing factors

Our analysis showed that establishing security governance procedures balancing control and autonomy of agile teams is crucial in LSAD. For this reason, we have examined the literature from our SLR and interview transcripts for influencing factors crucial to striking a suitable balance for individual situations, which completes our investigation of *RQ2*.

A. Product risk profile

It is crucial to consider the individual risk level of the software product, necessitating different activities and security requirements (I1, I3, I4, I6, I9), as "higher risk software [...] [has] increased test and compliance requirements prior to production" [67]. The grantable team autonomy highly depends on this factor. Lower risks may allow more freedom during implementation, while a higher risk profile might require teams

to adhere to certain boundaries, or even demand specific requirements (I6). For example, Microsoft [64] recommends regularly analyzing software components to determine risk levels, adjust security expert involvement and select security activities as required.

B. Context and risk appetite

Regulatory requirements have a substantial impact on grantable autonomy. In more security-critical industries, more rigidly enforced controls are required. The company culture and priorities also impact the resulting risk appetite, which might impact the level of team autonomy (I1, I4, I9). Company size also plays a role, as larger organizations tend to have more rigid governance structures and standards in place (I3).

C. Team-related factors

The concept of team security maturity follows the idea that "[higher] maturity leads to more autonomy" [60]. High team maturity might allow agile teams to release and deploy software without involving any other role, thereby reducing bottlenecks [50], [60]. The team's security maturity can also evolve during a project; thus, there is a need to adapt formal controls over time (I3). Team maturity can be assessed by methods such as previous security performance, self-assessments, and application security metrics (I4, I9).

D. Infrastructure and automation

Teams utilizing a standardized infrastructure and development pipeline reduce the need for manual security reviews (I2, I5) and enable the reuse of documentation for standardized components, further diminishing the need for manual controls (I4). Additionally, a standardized build and deployment pipeline can enforce automated security testing and quality gates, decreasing manual intervention and enhancing team autonomy [60] (I1, I6). Integrating automated security testing, such as software composition analysis as well as static and dynamic application security testing, allows even less experienced developers to contribute to application security (I9).

E. Additional constraints

Budget and time schedules, which impact the configuration of roles and activities, can also influence team autonomy (I2, I9).

VIII. Discussion

A. Key findings

Our results from examining the current state of security integration in LSAD show that there is a strong demand for security integration in LSAD environments, but organizations still face several crucial challenges. They are caused by the inherent tension between security and agility, as well as the peculiarities of LSAD implementations. In addition, the combination of security and LSAD introduces unique challenges, such as ambiguities regarding security-related roles, responsibilities, requirements, the adequate integration and usage of security testing tools, or, generally, the proper implementation of security activities in iterative development at scale. Given these issues, there is a need to clarify how organizations can best integrate security in LSAD.

Our analysis shows that most established scaling agile frameworks currently do not provide sufficient solutions to these challenges, except for DA and SAFe, which already provide helpful guidance. For example, DA suggests actions for raising security awareness (e.g., using guidelines), introducing security roles, the importance of testing and security assurance, and that security activities need to be integrated into iterative development. Also, DA stresses the mismatch between traditional governance and agile principles and emphasizes the importance of governance as an enablement function and less of a command and control instrument.

However, one could also argue that LSAD frameworks are not the right way to integrate these solutions because frameworks should not be too prescriptive [11]. Therefore, we identified initial promising solutions in the literature that partly also extend existing scaling agile frameworks. Although there is so far only little research on security integration in LSAD, insights from small-scale agile development and general quality approaches for LSAD can complement integrating security governance and compliance in LSAD. The identified influencing factors can further contribute to mitigate challenges by balancing control and autonomy of agile teams at scale. Nevertheless, there are no solutions yet that combine and leverage these solution approaches and factors, resulting in a potential for valuable further research.

In general, the interviewed industry experts confirmed the high practical relevance of the topic area and the need for further research. One specific aspect mentioned is the importance of investigating further how security-related governance mechanisms shift to agile teams and how much control will remain with central security governance units. Future research could build on the identified existing approaches and take the presented influencing factors into account. The identified challenges could be leveraged to systematically evaluate the impact of new solution approaches. Lastly, exploring the benefits of LSAD for security, for instance, to persuade governance and audit roles of the merits of autonomous teams, might be a valuable avenue for future research.

B. Limitations

In the following, we address potential limitations and countermeasures of our two primary research methods. Zhou et al. [68] identify validity threats for SLRs in software engineering, such as unsuitable search terms, databases, and inadequate inclusion and exclusion criteria. We aimed to ensure construct and internal validity through a robust and systematic selection of databases, search terms and publications. To address publication bias, which could affect the validity of results [30], we included gray literature as a known countermeasure [69]. The interview study's validity may be limited by interviewee selection and researcher influence, affecting objectivity during data collection and analysis [39]. We tried to mitigate this by following established guidelines of interview studies, such as

employing a semi-structured interview questionnaire, applying cyclic coding, and transcribing interviews to reassess discussed aspects, limiting researcher-inherent bias. To achieve external validity, we aimed for a diverse selection of experts and connected the findings from the interview study with insights from the literature. However, additional interviews could further enhance generalizability.

IX. Conclusion

Although agile methods are increasingly applied at scale, security risks have concurrently become increasingly frequent and complex. It is therefore crucial to better understand current challenges and solution approaches for integrating security governance and compliance in LSAD environments. We conducted a SLR and an interview study with nine experts. Answering our first RQ, we identified 15 relevant challenges, including those unique to LSAD and security, as well as others stemming from the inherent tension of agile development and security or from general scaling agile issues. The challenges, for example, address the lack of security integration within agile approaches, missing security awareness, unclear security-related roles and responsibilities, or the tension between agile-favored autonomy and compliance-required governance and control. Our framework analysis as part of answering our second RQ shows that most established scaling agile frameworks do not tackle the topic nor provide solution approaches. However, SAFe and DA recognize the need to consider security, governance, and compliance. Additionally, we presented solution approaches from literature and practice and described five factors that can be used to increase or decrease the autonomy of agile teams depending on a specific context, thereby mitigating some of the identified security governance and compliance challenges in LSAD. We advocate for the integration and assessment of these factors, allowing for increased autonomy in mature, capable teams, thus optimizing governance and control mechanisms to their minimum viable extent. Future research could delve deeper into this concept, suggesting explicit methods for autonomy assessment and resulting adaptions to governance processes, and evaluating the outcomes within LSAD environments.

References

[1] Y. Greer, Des; Hamon, "Agile software development," *Software: Practice and Experience*, vol. 41, no. 9, pp. 943–944, 2011.

[2] D. J. Anderson, *Agile management for software engineering: Applying the theory of constraints for business results*. Upper Saddle River, NJ, USA: Prentice Hall, 2004.

[3] T. Dingsøyr and N. B. Moe, "Towards principles of large-scale agile development," in *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*. Cham: Springer, 2014, vol. 199, pp. 1–8.

[4] A. Mordi and M. Schoop, "Scaling with an agile mindset - a conceptual approach to large-scale agile," in *AMCIS 2021 Proceedings*, 2021.

[5] Ö. Uludağ, P. Philipp, A. Putta, M. Paasivaara, C. Lassenius, and F. Matthes, "Revealing the state of the art of large-scale agile development research: A systematic mapping study," *Journal of Systems and Software*, vol. 194, p. 111473, 2022.

[6] E. van Veenendaal, "Next-generation software testers: Broaden or specialize!" in *The Future of Software Quality Assurance*, S. Goericke, Ed. Cham, Switzerland: Springer, 2020, pp. 229–243.

[7] M. Kalenda, P. Hyna, and B. Rossi, "Scaling agile in large organizations: Practices, challenges, and success factors," *Journal of Software: Evolution and Process*, vol. 30, no. 10, p. p. e1954, 2018.

[8] K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: A systematic literature review," *Journal of Systems and Software*, vol. 119, pp. 87–108, 2016.

[9] N. Newton, C. Anslow, and A. Drechsler, "Information security in agile software development projects: A critical success factor perspective," in *Proceedings of the 27th European Conference on Information Systems (ECIS)*, 2019, pp. 1 – 17.

[10] AON, "Making better decisions in uncertain times - aon's 2022 executive risk survey," 2022. [Online]. Available: https://www.aon.com/getmedia/683e7940-86d0-4fee-9b78-0b6330458d80/aon-2022-executive-risk-survey-report.pdf

[11] A. Poller, L. Kocksch, S. Türpe, F. A. Epp, and K. Kinder-Kurlanda, "Can security become a routine?" in *CSCW '17: Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. New York, NY, USA: ACM, 2017, pp. 2489–2503.

[12] S. Dännart, F. Moyón Constante, and K. Beckers, "An assessment model for continuous security compliance in large scale agile environments," in *Advanced Information Systems Engineering*. Cham, Switzerland: Springer, 2019, pp. 529–544.

[13] L. Bell, J. Bird, M. Brunton-Spall, and R. Smith, *Agile application security: Enabling security in a continuous delivery pipeline*. Sebastopol, CA: O'Reilly Media, 2017.

[14] A. van der Heijden, C. Broasca, and A. Serebrenik, "An empirical perspective on security challenges in large-scale agile software development," in *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, M. Oivo, Ed. New York, NY: ACM, 2018, pp. 1–4.

[15] D. Baca and B. Carlsson, "Agile development with security engineering activities," in *Proceeding of the 2nd workshop on Software engineering for sensor network applications - SESENA '11*. New York, New York, USA: ACM Press, 2011, pp. 149–158.

[16] S. Nägele, J.-P. Watzelt, and F. Matthes, "Investigating the current state of security in large-scale agile development," in *Agile Processes in Software Engineering and Extreme Programming: 23rd International Conference on Agile Software Development, XP 2022, Copenhagen, Denmark*. Springer, 2022, pp. 203–219.

[17] A. Poth, M. Kottke, C. Heimann, and A. Riel, "The efis framework for leveraging agile organizations within large enterprises," in *Agile Processes in Software Engineering and Extreme Programming – Workshops*, P. Gregory and P. Kruchten, Eds., vol. 426. Cham: Springer International Publishing, 2021, pp. 42–51.

[18] R. Knaster and D. Leffingwell, *SAFe 5.0 Distilled: Achieving Business Agility with the Scaled Agile Framework*. Boston: Addison-Wesley, 2020.

[19] S. W. Ambler and M. Lines, *An Executive's Guide to Disciplined Agile: Winning the Race to Business Agility*. North Charleston, SC, United States: CreateSpace Independent Publishing Platform, 2017.

[20] J. Sutherland and Scrum Inc., "The scrum@scale® guide: The definitive guide to the scrum@scale framework," 2021. [Online]. Available: https://www.scrumatscale.com/wp-content/uploads/2020/12/official-scrum-at-scale-guide.pdf

[21] J.-P. Steghöfer, E. Knauss, J. Horkoff, and R. Wohlrab, "Challenges of scaled agile for safety-critical systems," in *Product-Focused Software Process Improvement*. Cham, Switzerland: Springer, 2019, pp. 350–366.

[22] H. Edison, X. Wang, and K. Conboy, "Comparing methods for large-scale agile software development: A systematic literature review," *IEEE Transactions on Software Engineering*, pp. 1–23, 2021.

[23] P. Williams, "Information security governance," *Information Security Technical Report*, vol. 6, no. 3, pp. 60–70, 2001.

[24] S. H. Solms and R. Solms, *Information Security Governance*. Boston, MA: Springer US, 2009.

[25] K. Julisch, "Security compliance," in *Proceedings of the 2008 workshop on New security paradigms*, M. Bishop, Ed. New York, NY: ACM, 2008, pp. 71–74.

[26] B. Blakley, E. McDermott, and D. Geer, "Information security is information risk management," in *Proceedings of the 2001 workshop on New security paradigms*, V. Raskin, S. J. Greenwald, B. Timmerman, and D. Kienzle, Eds. New York, NY: ACM, 2001, pp. 97–104.

[27] A. Da Veiga and J. Eloff, "An information security governance framework," *Information Systems Management*, vol. 24, pp. 361–372, 2007.

[28] J. Webster and R. T. Watson, "Analyzing the past to prepare for the future: Writing a literature review," *MIS Quarterly*, pp. xiii–xxiii, 2002.

[29] J. Brocke, A. Simons, B. Niehaves, B. Niehaves, K. Reimer, R. Plattfaut, and A. Cleven, "Reconstructing the giant: On the importance of rigour in documenting the literature search process," *ECIS 2009 Proceedings*, 2009. [Online]. Available: https://aisel.aisnet.org/ecis2009/161

[30] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," School of Computer Science and Mathematics, Keele University, Keele, Staffs, United Kingdom, Tech. Rep., 01 2007.

[31] M. D. Myers and M. Newman, "The qualitative interview in is research: Examining the craft," *Information and Organization*, vol. 17, no. 1, pp. 2–26, 2007.

[32] H. J. Rubin and I. S. Rubin, *Qualitative Interviewing: The Art of Hearing Data*, 3rd ed. SAGE, 2011.

[33] J. Saldaña, *The coding manual for qualitative researchers*, 3rd ed. Los Angeles and London: SAGE, 2016.

[34] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007.

[35] V. Garousi, M. Felderer, and M. V. Mäntylä, "Guidelines for including grey literature and conducting multivocal literature reviews in software engineering," *Information and Software Technology*, vol. 106, pp. 101–121, 2019.

[36] Y. Levy and T. J. Ellis, "A systems approach to conduct an effective literature review in support of information systems research," *Informing Science: The International Journal of an Emerging Transdiscipline*, vol. 9, pp. 181–212, 2006.

[37] K. Beck et al., "Manifesto for agile software development," 2001. [Online]. Available: https://agilemanifesto.org

[38] V. Garousi, M. Felderer, and M. V. Mäntylä, "The need for multivocal literature reviews in software engineering," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*. New York, NY, USA: ACM, 2016, pp. 1–6.

[39] M. B. Miles, A. M. Huberman, and J. Saldaña, *Qualitative data analysis: A methods sourcebook*, 3rd ed. Thousand Oaks Californinia: SAGE Publications Inc, 2014.

[40] F. Moyón, D. Méndez, K. Beckers, and S. Klepper, "How to integrate security compliance requirements with agile software engineering at scale?" in *Product-Focused Software Process Improvement*. Cham, Switzerland: Springer, 2020, pp. 69–87.

[41] S. Türpe and A. Poller, "Managing security work in scrum: Tensions and challenges," in *SecSE@ESORICS 2017*, 2017. [Online]. Available: http://ceur-ws.org/Vol-1977/paper4.pdf

[42] S. Bartsch, "Practitioners' perspectives on security in agile development," in *2011 Sixth International Conference on Availability, Reliability and Security (ARES 2011)*. Piscataway, NJ: IEEE, 2011, pp. 479–484.

[43] D. Baca, M. Boldt, B. Carlsson, and A. Jacobsson, "A novel security-enhanced agile software development process applied in an industrial setting," in *2015 10th International Conference on Availability, Reliability and Security*, 2015, pp. 11–19.

[44] K. Goertzel, T. Winograd, H. McKinley, L. Oh, M. Colon, T. McGibbon, E. Fedchak, and R. Vienneau, "Software security assurance: A state-of-art report (sar)," Fort Belvoir, VA. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA472363.pdf

[45] SAFECode, "Practical security stories and security tasks," 2012. [Online]. Available: https://safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf

[46] M. Felderer and I. Pekaric, "Research challenges in empowering agile teams with security knowledge based on public and private information sources," in *22nd European Symposium on Research in Computer Security (ESORICS), Oslo, Norway, 2017*, vol. 1977, 2017, pp. 1–7.

[47] K. Beznosov and P. Kruchten, "Towards agile security assurance," in *Proceedings of the 2004 workshop on New security paradigms*, V. Raskin, Ed. New York, NY: ACM, 2004, pp. 47–54.

[48] S. Nilsson Tengstrand, P. Tomaszewski, M. Borg, and R. Jabangwe, "Challenges of adopting safe in the banking industry – a study two years after its introduction," in *Agile Processes in Software Engineering and Extreme Programming*, vol. 419. Cham: Springer International Publishing, 2021, pp. 157–171.

[49] A. Putta, M. Paasivaara, and C. Lassenius, "Benefits and challenges of adopting the scaled agile framework (safe): Preliminary results from a multivocal literature review," in *Product-Focused Software Process Improvement*, vol. 11271. Cham: Springer International Publishing, 2018, pp. 334–351.

[50] Y. Petit and C. Marnewick, "Earn your wings: A novel approach to deployment governance," in *Agile Processes in Software Engineering and Extreme Programming – Workshops*, R. Hoda, Ed., vol. 364. Cham, Switzerland: Springer, 2019, pp. 64–71.

[51] L.-O. Damm, "How is the software development process impacted when a large company goes agile?" in *Proceedings of the 2015 International Conference on Software and System Process*. New York, NY, USA: ACM, 2015, pp. 6–7.

[52] R. Naidoo and S. Rikhsoto, "Balancing autonomy and control tensions in large-scale agile," in *Information Systems Development: Crossing Boundaries between Development and Operations (DevOps) in Information Systems (ISD2021 Proceedings)*. Valencia, Spain: Universitat Politècnica de València, 2021.

[53] A. Šāblis and D. Šmite, "Agile teams in large-scale distributed context," in *Proceedings of the Scientific Workshop Proceedings of XP2016*. New York, NY, USA: ACM, 2016, pp. 1–5.

[54] W. Alsaqaf, M. Daneva, and R. Wieringa, "Quality requirements challenges in the context of large-scale distributed agile: An empirical study," in *Requirements Engineering: Foundation for Software Quality*. Cham: Springer International Publishing, 2018, pp. 139–154.

[55] Scrum.org, "The nexus™ guide: The definitive guide to scaling scrum with nexus," 2021. [Online]. Available: https://scrumorg-website-prod.s3.amazonaws.com/drupal/2021-01/NexusGuide%202021_0.pdf

[56] C. Larman and B. Vodde, *Large-scale Scrum: More with LeSS*. Boston and Amsterdam and London: Addison-Wesley, 2016.

[57] H. Kniberg and A. Ivarsson, "Scaling agile @ spotify with tribes, squads, chapters & guilds," 2012. [Online]. Available: https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf

[58] S. W. Ambler and M. Lines, *Choose Your WoW - First Edition: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working*, 1st ed. Project Management Institute, 2020.

[59] F. Moyon, K. Beckers, S. Klepper, P. Lachberger, and B. Bruegge, "Towards continuous security compliance in agile software development at scale," in *2018 ACM/IEEE 4th International Workshop on Rapid Continuous Software Engineering (RCoSE 2018)*, 2018, pp. 31–34.

[60] A. Poth, J. Jacobsen, and A. Riel, "Systematic agile development in regulated environments," in *Systems, Software and Services Process Improvement*, M. Yilmaz, J. Niemann, P. Clarke, and R. Messnarz, Eds., vol. 1251. Cham: Springer, 2020, pp. 191–202.

[61] M. Boldt, A. Jacobsson, D. Baca, and B. Carlsson, "Introducing a novel security-enhanced agile software development process," *International Journal of Secure Software Engineering*, vol. 8, no. 2, pp. 26–52, 2017.

[62] G. K. Hanssen, B. Haugset, T. Stålhane, T. Myklebust, and I. Kulbrandstad, "Quality assurance in scrum applied to safety critical software," in *Agile Processes, in Software Engineering, and Extreme Programming*, vol. 251. Cham: Springer International Publishing, 2016, pp. 92–103.

[63] B. Fitzgerald, K.-J. Stol, R. O'Sullivan, and D. O'Brien, "Scaling agile methods to regulated environments: An industry case study," in *35th International Conference on Software Engineering (ICSE)*, 2013, pp. 863–872.

[64] Microsoft, "Security development lifecycle - sdl process guidance version 5.2," 2012. [Online]. Available: https://www.microsoft.com/en-us/download/details.aspx?id=29884

[65] B. Sullivan, "Agile sdl: Streamline security practices for agile development," 2019. [Online]. Available: https://docs.microsoft.com/en-us/archive/msdn-magazine/2008/november/agile-sdl-streamline-security-practices-for-agile-development

[66] OWASP and R. van der Veer, "Samm agile guidance," 2022. [Online]. Available: https://owaspsamm.org/guidance/agile/

[67] SAFECode, "Software security takes a champion: A short guide on building and sustaining a successful security champions program." [Online]. Available: http://safecode.org/wp-content/uploads/2019/02/Security-Champions-2019-.pdf

[68] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang, "A map of threats to validity of systematic literature reviews in software engineering," in *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2016, pp. 153–160.

[69] A. Yasin, R. Fatima, L. Wen, W. Afzal, M. Azhar, and R. Torkar, "On using grey literature and google scholar in systematic literature reviews in software engineering," *IEEE Access*, vol. 8, pp. 36 226–36 243, 2020.

Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry

Sascha Nägele
sascha.naegele@tum.de
Technical University Munich
Munich, Germany

Lorena Korn
lorena.korn@tum.de
Technical University Munich
Munich, Germany

Florian Matthes
matthes@tum.de
Technical University Munich
Munich, Germany

ABSTRACT

Agile development methods have pervaded software engineering and are increasingly applied in large projects and organizations. At the same time, security threats and restrictive legislation regarding security and privacy are steadily rising. These two trends of agile software development at scale and increasingly important security requirements are often at odds with each other. Academic literature widely acknowledges the challenges therefrom and discusses approaches to integrate these two partly conflicting trends. However, several researchers point out a need for empirical studies and evaluations of these approaches in practice. To fill this research gap, we conducted a case study in the finance industry. We identified 27 agile security approaches in academic literature. Based on these theoretical findings, we carried out observations, document analysis, and unstructured interviews to identify which approaches the case company applies. We then conducted semi-structured interviews with 10 experts and a survey with 62 participants to evaluate 14 approaches. One of the key results is that role and knowledge approaches, such as dedicated security roles and communities, are especially important in scaled agile development environments. In addition, the most beneficial security activities are easy-to-integrate, such as a security tagging system, peer security code reviews, security stories, and threat poker. We also contribute evaluation criteria as well as drivers and obstacles for the adoption of agile security approaches that can be used for further research and practice.

CCS CONCEPTS

• **Security and privacy → Software security engineering**.

KEYWORDS

large-scale agile, security, software development, case study

ACM Reference Format:
Sascha Nägele, Lorena Korn, and Florian Matthes. 2023. Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry. In *The 18th International Conference on Availability, Reliability and Security (ARES 2023), August 29–September 01, 2023, Benevento, Italy*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3600160.3600170

ARES 2023, August 29–September 01, 2023, Benevento, Italy

ACM ISBN 979-8-4007-0772-8/23/08…$15.00
https://doi.org/10.1145/3600160.3600170

1 INTRODUCTION

Initially emerging in the late 1990s and captured in the Agile Manifesto [8], agile development approaches are predominant in software engineering today [27, 53] and of broad interest in academic research [20]. In addition, agile development methods are increasingly applied to large projects and companies with many software development teams working together [17, 19, 56]. Companies thereby hope to benefit from the advantages of these methods, such as enhanced adaptability to fast-evolving environments and accelerated time to market [42].

At the same time, a rise in cybercrime puts companies' business success at risk. According to Check Point Research, global cyberattacks rose by 38% in 2022 compared to 2021 [44]. The Covid-19 pandemic further increases the complexity and growing number of cyber attacks as changing work conditions raise the ubiquity and dependence on information technology [23]. In addition, increased restrictive legislation concerning security and privacy has required securing software engineering processes and products [47].

Nevertheless, the prioritization of functionality in user-centered, iterative agile approaches often overshadows security concerns [5, 40]. Conversely, the need for security compliance and traceability of security requirements hinders early and frequent software delivery [7, 11, 37]. Due to additional regulatory requirements, this is especially challenging when developing security-critical systems in regulated industries such as finance and healthcare [22, 26, 29, 51].

To address this conflict and achieve security in large-scale agile software development (LSAD), security activities must be adapted to agile environments and evaluated in practice. Scientific literature already discussed integrating security into large-scale agile practices [2, 3, 31]. However, several researchers point to a lack of empirical studies and evaluations in industrial practice [12, 37, 39, 46]. According to Rindell et al. [46], "empirical evidence is sporadic and largely incomplete." Bishop and Rowland [12] stress that "there is further investigation needed [...] in regard to empirical evaluations of the proposed agile and secure software development integration approaches." Therefore, we conducted a single-case study to contribute to the empirical evidence.

The three **research questions** we seek to answer are:

- RQ1: Which agile security approaches does the case company use in its scaled agile development environment?
- RQ2: What are the driving factors and main obstacles for adopting these approaches?
- RQ3: How are existing and potentially suitable approaches evaluated?

We contribute detailed insights to these questions by presenting results of observations, document analysis, interviews, and a survey in the case company.

2 BACKGROUND AND RELATED WORK

When using the term security, we refer to the concepts of information security and software security. *Information security* is concerned with protecting the confidentiality, integrity, and availability of information [9, 28]. McGraw [36] defines *software security* as "the process of designing, building, and testing software for security." Software security aims to build software and applications free from vulnerabilities that are highly robust, recover quickly, limit harm, and continue to operate correctly in case of attacks [1, 36].

Various definitions for LSAD exist, specifying scale based on a minimum number of teams and people involved [17, 18]. Dingsøyr et al. [18] define the term large-scale as a collaboration of two to nine agile teams of a maximum of nine developers. According to their taxonomy, a software development project with more than 10 collaborating teams is very large-scale [18]. Dikert et al. [17] state that large-scale development requires "50 or more people or at least six teams." The large-scale environment of our case company meets these definitions.

Scientific research discusses multiple approaches for embedding security in agile development environments [10, 38, 39, 46]. In this paper, we call these approaches *agile security approaches* and define them as activities, practices, or strategies that enhance software security or information security, improve security governance or security compliance and mitigate challenges between agile methods and security in LSAD. Using a systematic literature review, we extracted 27 such approaches from 60 relevant publications, of which we found 11 in our case company. Chapter 4 describes those adopted approaches in more detail.

Due to similarity with our research goals and methods, we consider the following studies as related work.

Bartsch [6] aims to enhance the understanding of information security in agile software projects from a practitioner's perspective. He empirically examines identified challenges and mitigation strategies through 10 semi-structured interviews, mainly conducted with agile developers. Bezerra et al. [10] seek to create security policies containing security approaches suitable for agile software engineering. These policies aim to help agile teams adopt security approaches by explaining how, when, and who should use them. The results are based on a literature review, workshops, and expert interviews. Newton et al. [39] strive to determine critical success factors for facilitating information security in agile development environments by comparing academic and practically adopted agile security approaches. For future research, the authors suggest empirical studies that "expand the data collection further across organizations in different countries, of different sizes, maturity levels, industries, or business models" [39]. We aim to contribute to this body of existing research with our study.

3 RESEARCH METHODOLOGY

The primary applied research method is a case study conducted for six months in a large organization in the finance industry. We base our approach on Runeson and Höst's [49] process for conducting case studies, complemented by the methodological works of Yin [55] and Eisenhardt [21]. Our method comprises of two main steps: (i) study design, and (ii) data collection and analysis. We chose to carry out a single-case study because it is highly suitable for examining present phenomena in a real-world context [55], the effects of a transforming event [49], and the reasons for that event [21]. In our work, agile security approaches are the studied phenomena. We inspect the driving factors, obstacles, and the advantages and drawbacks of their adoption. Furthermore, a case study is a suitable approach to explore a research topic in an area where empirical evidence is scarce [21]. We thereby aim to expand the empirical body of knowledge and contribute a novel industry perspective to existing research.

3.1 Case study design

At first, we planned the design of our case study, including its objective, study object, theory basis, and applied methods [49].

3.1.1 Case study objective. The overall objective of this case study is to explore the adoption of security approaches in regulated LSAD. Specifically, we aim to answer the three research questions presented in the introduction.

3.1.2 Case study object. The investigated case is one of the world's largest financial organizations. We selected this case because of two reasons. First, we want to augment existing research by adding empirical evidence on agile security approaches in regulated sectors. Secondly, due to high regulatory pressure and their recent expansion of agile development methods at large-scale, the tensions between agile methods and security are particularly apparent in the target organization. Due to the sensitive nature of our research topic, we keep the organization anonymous and refrain from providing details that make the company easily identifiable. Nevertheless, Section 4.1 provides a detailed case description explaining important context for this study. The units of analysis are mainly the agile development teams and a central security department of the company.

3.1.3 Theory basis. Section 2 contains the theory that builds the basis and clarifies the context of this case study. It provides an overview of information security, LSAD, and related work.

3.1.4 Applied methods. A case study should involve distinct methods for data collection and rely on multiple data sources to cover different viewpoints and to achieve a more robust outcome [49, 55]. We used direct and indirect methods for data collection during our case study. By combining these methods, we obtain qualitative and quantitative evidence that can be "highly synergistic", according to Eisenhardt [21].

The following subsection provides a detailed description of the applied data collection methods.

3.2 Data collection and analysis methods

The second step of our approach is selecting and specifying the direct and indirect methods used for data collection and analysis during our case study. We used the following sources of evidence to collect data directly:

- Survey
- Unstructured interviews
- Semi-structured interviews

Moreover, we supplemented those with the following indirect data collection methods:

- Observations
- Documentation and archival records

We employ these methods to understand the case study company's circumstances as thoroughly as possible and to identify all security activities used in their LSAD environment (RQ1). We use the survey and expert interviews primarily to evaluate empirically the success in the adoption and potential suitability of preselected agile security approaches (RQ2, RQ3). The following subsections provide more details regarding the methods used.

3.2.1 Observation, unstructured interviews, and document analysis. Throughout the six months of our case study, we continuously analyzed documents and archival records relevant to our research scope, such as topic-specific or corporate presentations and files, intranet articles, or internal reports. In addition, we participated in numerous meetings to collect data through observations, mainly in the security department and agile teams of the examined organization. Every week, we conducted unstructured interviews and informal conversations with roles related to security or LSAD.

3.2.2 Survey. We created and conducted the survey based on the methods of Fowler [24]. According to Fowler, the three fundamental elements for "good survey design" are sampling, question design and data collection [24].

Sampling. Sampling is the selection of "a small subset of a population representative for a whole population" [24]. The target population of the survey were individuals of the agile structure of the case study organization. We especially targeted so-called Security Champions (SCs) since they represent one agile security role approach and are most involved in security activities. In total, we received 62 valid responses of which about one third were from SCs.

Question design. We designed the questionnaire based on the recommendations of Fowler [24]. Overall, our survey included 29 questions relevant to our research goals. We tailored some of the questions to specific roles, i.e., participants only received certain questions if they had selected the corresponding roles at the beginning of the questionnaire. The majority of questions are closed questions to ensure the comparability of answers. For measuring the responses, we used an ordinal, five-category scale [24]. In some cases, we added a not applicable option. In addition, several questions were followed by free-text fields, allowing respondents to provide additional qualitative feedback. Prior to publication, a pilot group of 10 individuals tested the questionnaire. Based on their feedback, we made some adjustments to the response options and wording of the questions to improve reliability and construct validity [24].

Data collection. We chose a self-administered, anonymous online survey to achieve maximum returns [24]. The survey design and measurement instrument we used is TemboSocial [52]. We first announced the survey in an information session for agile teams and emailed the survey's URL to the attendees. In addition, we distributed it through an article in the company's social network and through the security channel in the company messenger. We also asked the recipients to share the survey link with their colleagues and agile team members. The survey remained open for five weeks. We further analyzed the survey results in Excel based on an exported data file.

3.2.3 Expert interviews. For our four-step expert interview approach, we utilize the methodological works of Bogner [13], Bortz and Döring [14], Kvale [33], and Mayring [35].

Operationalization. We transferred the evaluation's goals into a semi-structured interview guide with 26 questions, considering the rules of good expert interviews [14, 33]. Table 1 describes the 12 evaluation criteria we established as part of the interview guide. We used the results of a literature review we conducted prior to this study to select these criteria. They allowed for structurally inquiring the approaches' compatibility within the regulated LSAD environment of the examined organization.

Expert selection. We derived the selection of interview partners from the expert definition of Bogner [13]. The most common role in our interviews is the team-external security expert, providing the most insight from first-hand experience with multiple teams. The average experience of our experts in security is four years, and in LSAD four and a half years.

Interview conduction. The interview preparation included a one-hour preliminary meeting that we used to introduce the evaluation goals and security approaches, as well as to answer any questions. The approaches were briefly defined and explained, both in general and with respect to their adoption in the case company, with the aim of achieving a common understanding among the experts. The interviews took an average of one hour and 20 minutes per interviewee. We adjusted the guiding questions to follow up the interviewees' statements about their favored approaches [33]. In addition, we summarized the key aspects mentioned by the experts during the interview to get their feedback and avoid misunderstandings [49]. As proposed by Kvale [33], we carried out as many interviews until we reached a "point of satisfaction, where further interviews yielded little new knowledge." With the consent of the interviewees, we recorded all interviews for the purpose of transcription. We ensured that the results were only used in anonymized form, as this was assured to the interviewees beforehand.

Interview analysis. To analyze the qualitative data obtained from the semi-structured interviews, we performed the systematic qualitative content analysis methods of Mayring [35]. We first manually created full transcripts of all expert interviews' audio recordings, followed by a summarizing content analysis [35] for interview coding. Using the coding methods of paraphrasing, generalization, and reduction, we extracted and systematically structured the most relevant interview segments for assessing the agile security approaches.

3.2.4 Preselection and evaluation criteria. Due to the limited time frame for the case study, the availability of interviewees, and the large number of identified security approaches, we narrowed down the number of approaches to be evaluated prior to conducting the survey and expert interviews. From the 27 approaches we identified in the literature, we excluded those with less than 10 references, resulting in 16 approaches, e.g., security training (24 references), team-internal security expert (11 references), and team-external security role (10 references). We further filtered publications due

Table 1: Evaluation criteria for agile security approaches

Agile criteria [25, 30, 50]	
Iterative	For example, in each iteration or regularly in other time intervals.
Adaptable	Adaptive and flexible to changing requirements.
Interactive	Focused on people and their interaction.
Lean	Accelerating software delivery, maximizing resource utilization or reducing *waste*, such as unnecessary documentation effort.
Simple	Easy to understand and easy to use.
Motivating	For example, by using fun components.
Large-scale criteria [17, 19]	
Autonomy	Increasing the autonomy of agile teams, for example, through increased security knowledge.
Collaboration	Improving cross-team collaboration and coordination between agile teams.
Knowledge exchange	Enhancing security knowledge exchange between agile teams.
Technical consistency	Promoting technical consistency across agile teams and the reduction of technical debt such as risk acceptances.
Regulatory criteria [22, 26]	
Traceability	Fostering the traceability of regulatory and company-internal security requirements.
Compliance	Facilitating compliance with regulatory and company-internal security guidelines.

to a lack of concreteness, meaning the approach descriptions did not provide enough level of detail to be evaluated in a meaningful way. In total, we evaluated 14 of the 27 agile information security approaches identified in relevant literature.

4 ADOPTED AGILE SECURITY APPROACHES

This chapter presents the initial results of our case study. To set the stage, we provide a case description. We then present the identified security approaches used in the LSAD environment of the case company, followed by the discovered adoption drivers and obstacles.

4.1 Case description

The case under investigation is a large company in the finance industry. The studied organization has a large-scale development program, with over 1,500 individuals active in their agile teams. Security is crucial for the analyzed organization for several reasons, e.g., protecting customer and employee information is critical for the company's success, and companies in the finance industry are subject to strict regulations. As we typically observe in large organizations, governance functions such as security or privacy are separate departments and stakeholders to software development teams. The security department includes governance, compliance, and technical roles. The primary role of the governance team is the security officer, who is responsible for activities such as establishing internal policies, ensuring compliance with external regulations, security awareness, audits, and application security verification. The technical team comprises security analysts and ethical hackers concerned with security and penetration testing, vulnerability management, incident response, red teaming, and forensics. In recent years, the company has started an initiative to transform its software development from using traditional development approaches (e.g., waterfall model) with less frequent releases into more incremental, agile working methods. The Spotify model [32] partly inspires the LSAD framework of the case study organization. Each transformed development project is steered by business sponsors and consists of several tribes that comprise multiple teams, i.e., squads. The tribe's teams are cross-functional and have a size of about three to 12 members, consisting of a product owner, a scrum master, developers, and business analysts. Development occurs in sprints, which are typically divided into longer periods with specific goals. For example, three months might include six bi-weekly iterations, providing a broader planning horizon. Due to the goal of self-organizing teams, we found that software engineers apply a combination of practices from lean and agile methods such as Scrum, Extreme Programming, Kanban, and Test Driven Development, whereby the Scrum framework is predominant. In addition, the case company established multiple types of guilds, as described in the Spotify model [32]. Guilds unite individuals from agile teams and governance functions to engage on focus topics. Examples for these topics are security, software architecture, or user experience. These guilds aim to provide a form of lean governance by enabling knowledge sharing to empower teams to better comply with guidelines and standards. Furthermore, going beyond that, teams should also be able to coin the further development of these standards.

4.2 Adopted roles and knowledge approaches

We found four agile security approaches concerning roles and knowledge in the investigated case company. The security guild drives the adoption of these approaches. It focuses on the sustainable integration of security in agile methods and the establishment of security as a self-evident, ongoing activity in agile teams. It consists of three key roles: a security expert in the team, a team-external role, and a security officer. The guild aims to coordinate overarching technical security topics in the agile development teams. In addition, it strives to increase the agile developers' knowledge and awareness of security to enable independent security-related decisions by the teams. The guild facilitates security training and approaches for cross-team and cross-tribe security knowledge exchange to achieve this.

4.2.1 Security knowledge exchange. The case company established multiple approaches to share security knowledge. Examples include Security Engineers (SEs) hosting a monthly event to present current security guild topics. These include the latest security news, new support offerings, or introducing new members. In addition, SEs jointly discuss issues from their daily work at least weekly. These topics include cloud and mobile security, risk management, secure coding, and system hardening, amongst others, and are shared with the SCs to distribute the gained expertise to the development teams. In addition, security knowledge exchange occurs via dedicated security channels in the company-internal messenger and on the company-internal social network. This includes blog articles and

FAQs, e.g., about organizing penetration tests or changes to internal security policies.

4.2.2 Team-external security role. The case company introduced a team-external security role that advises and supports agile teams, in the following referred to as a SE. SEs possess not only profound specialized expertise in security but also have hands-on experience with agile development methodologies. Their main tasks are to advise and support agile teams on security issues like security guidelines, the implementation of security measures, risk analysis, and security documentation. SEs also introduce security experts in agile teams to their role and provide them with proactive assistance in the form of mentoring.

In addition, SEs support security training, foster security knowledge exchange, and create blueprints and best practices for agile developers concerning internal and regulatory security standards. Together with security officers, SEs translate high-level, strategic security requirements into operational guidelines and further develop and communicate internal security policies to agile teams.

The SEs take part in the agile development process by participating in regular, agile meetings of their teams, such as sprint planning or sprint review. They regularly discuss critical security issues with their development teams and prepare the evaluation and, if necessary, approval of security-relevant artifacts for new releases.

4.2.3 Team-internal security expert. In the examined organization, more than 55 agile team members have a designated additional role as a security expert, often referred to as a SC. A SC is interested in security topics, stays up-to-date on the latest security developments, and keeps track of security concerns in their team. SCs are responsible for creating security documentation, organizing product-related security measures and spreading security knowledge within the agile development teams. Furthermore, they support the product owner with risk analysis, compliance with regulations and guidelines, and collaborate with security officers. Although SCs critically examine their software product's security architecture, the SC does not fully take on the role of a security architect, as described in scientific literature [4, 16].

4.2.4 Security training. The security department offers basic security training for each employee. In addition, the SEs promote specific training courses for product owners, SCs and agile developers to enhance their security expertise and awareness. The training sessions are recorded and made available online for all other interested parties. Topics include for example, security fundamentals, tips for security documentation, vulnerability demos, frequent penetration test findings, security best practices, risk assessment, secure coding, and security testing. Furthermore, short security knowledge videos are published over the internal social network.

4.3 Adopted methodology approaches

In total, we observed seven security methodology approaches. However, these approaches have different degrees of maturity in their application, and some are still in the testing phase.

4.3.1 Security stories. In the case study company, a variant of the extended version of security stories, namely the identification component proposed by Pohl and Hof [43], is used. The goal is to scrutinize the security relevance of each user story in the product backlog and label it accordingly. The goal of the approach is to achieve a better overview of the security-related stories and issues in the agile teams and to raise the agile team members' awareness of security.

4.3.2 Security criteria in definition of done. In some cases, the agile teams in the case study company have adopted security criteria at the user story level. Security officers or SEs might support their definition or even request specific criteria for a user story.

4.3.3 Pair programming. Most of the development teams use pair programming as an integral practice in the company's agile transformation initiative. In the interest of security, pairing with a SC or SE can be very beneficial. However, at the time of our case study, pair programming is often not yet done with a focus on security.

4.3.4 Security meetings. Usually, advisory meetings between SEs and SCs take place in a frequency depending on the needs of the teams as well as the complexity and security relevance of the product. Since teams of a tribe often encounter similar issues, multiple SCs might join. Topics covered during these meetings mainly include current concerns of the development team, such as security architecture or penetration test reports. Mostly, the agile team members proactively contribute issues that emerge during the planning and development. A final security review meeting with a security officer might be necessary to obtain approval for critical software releases.

4.3.5 Security documentation artifacts. Development teams create and maintain various documentation artifacts with security relevance. Among these are (security) architecture diagrams, risk assessments and implemented measures, and backup and restore concepts. Some of the artifacts require a review by a SE or security officer, depending on the criticality. To assist, the SEs provide checklists and templates. The security department provides a central tool for regulatory-compliant storage of those artifacts.

4.3.6 Security testing. The extent of security testing differs from team to team and product criticality. The dominant adversarial security testing approach is penetration testing. Generally, a penetration test is required for all new applications and major updates that include security-relevant changes. The tests are organized by individual development teams and supported and executed by the technical team of the security department or an external contractor. Another established approach for agile teams to identify application vulnerabilities is participating in a bug bounty program. They offer continuous testing by ethical hackers who then receive rewards for the vulnerabilities found. This continuity is especially valuable in products with frequent, iterative releases. Teams track and fix detected vulnerabilities with the support of a SE. Depending on the criticality, SEs, SCs, or a contractor conduct re-tests to verify whether implemented solutions work as expected. Some agile teams also use dynamic application security testing tools as a more automated and autonomous approach to security testing. At the time of the case study, however, this was yet to be observed in a majority of teams.

Table 2: Drivers and obstacles in the adoption process of agile security approaches

Drivers	No. of references
Instrinsic motivation	5
Ease of approach	4
Clear benefit of approach	4
Similarity to known practices	2
Regulation or internal guidelines	2
Obstacles	**No. of references**
Lack of security awareness & knowledge	7
Need for more resources	6
Focus on functionality	5
Status of security in the organization & teams	4
Top-down vs. bottom-up driven adoption	4
Lack of agile security support	2

4.3.7 Security code review. Agile teams partly apply tools for static code analysis. However, the static code analysis is not exclusively focused on security aspects, but to achieve a clean, maintainable, and secure code basis by detecting code smells, bugs, and security vulnerabilities. There is no universal requirement for the frequency and amount of resolved issues, but each team decides based on their needs in consultation with a SE or security officer. Furthermore, some SEs started conducting manual security-related peer code reviews with their SCs.

4.4 Driving factors and obstacles in the adoption process

This section answers our second research question by outlining the 11 driving and hindering factors for adopting agile security approaches that we identified during the semi-structured interviews with 10 experts. Additionally, we compare them to the findings in relevant academic literature. Table 2 overviews the driving factors and obstacles in the adoption process and the number of interview partners who mentioned them.

4.4.1 Driving factors for adoption. The driving factor most frequently mentioned in the expert interviews is the intrinsic motivation of software developers. It might originate from enjoying an agile security approach or personal security interest. One SE outlined that teams often establish approaches because of individual effort and "less because the organization or their product owner demands it." Teams tend to apply those approaches voluntarily when they are convinced that the respective approach is valuable. Other agile teams act out of a desire to "make their product secure and protect the customer." However, this significantly differs between teams since intrinsic motivation varies substantially. In addition, adopting agile security approaches is facilitated by simplicity and low effort involved in introducing the approach. Experts agreed that it is better to start with approaches that agile teams can use "without a large increase in know-how."

Another key driver in the adoption process mentioned by several experts is clearly pointing out the benefit provided by the approaches to agile teams. According to one SE, teams will more likely adopt an agile security approach "if the added value is made clear." Another interviewee establishes a link to the self-motivation factor of agile teams: "If you understand the meaning behind it, you will do it." In addition, two experts describe the consideration of the perceived benefits compared to the effort. This corresponds to the finding of Keramati and Mirian-Hosseinabadi [30] that the introduction of security approaches in agile development environments depends on its cost-benefit ratio.

Another important driver is the similarity to the development teams' familiar agile or security practices. One interviewed expert stated that approaches building on well-established methods and knowledge tend to be more accepted than those that are conceptually completely new. Another participant emphasized that a security approach that integrates with or partially encompasses the tasks already carried out by agile teams has a greater chance of being implemented. Barbosa and Sampaio [5] also identified this driving factor and recommend using existing agile practices to promote the adoption process of security activities. Another driving force identified in both the expert interviews and academic literature [45, 48] are security regulations or internal security guidelines. An interview partner stated that agile teams are motivated to use an approach if it enables them to meet the security requirements needed for the software to be released more quickly. Another interviewee indicates that some development teams only apply security approaches due to mandatory internal policies.

4.4.2 Obstacles for adoption. Seven interviewed professionals perceive the lack of security awareness and knowledge as a main obstacle in the adoption process. Possible reasons given for this lack of awareness are the developers' missing experience with security and the limited prominence of agile security approaches. Also, other stakeholders such as the product owner, customer, or sponsors often lack understanding of security topics. In addition, two experts are convinced that team members with security expertise are the basis for adequately using security methods. Findings in scientific research acknowledge security awareness and competence as essential factors for implementing security approaches in agile development settings [41]. Another major impediment to adopting agile security approaches is the need for more resources, more precisely, budget and time. One of the experts stresses that sponsors and product owners have to allocate enough time to their agile teams to ensure the use of security approaches. When teams are already working at full capacity due to functional requirements, they often prioritize these over security improvements. Relevant literature also identified the expenditure of costs and time for the adoption process as a hindering factor [5]. The willingness to spend resources on agile security approaches depends on the status and attitude towards security in the respective organization and development teams. The experts reported that security played a secondary role at the beginning of the agile transformation. However, it has improved over time because of the measures taken, e.g., through security roles. Furthermore, in some agile teams, the attitude towards security is biased as some team members have had negative experiences, e.g., release delays because of security checks. This might lead to frustration, potentially resulting in the urge to bypass certain security practices in order to avoid getting slowed down. In addition, it might negatively affect the willingness to try out and

adopt security approaches. On the other hand, experts stressed that this circumstance highlights the importance of security approaches that integrate well with LSAD environments. Another obstacle to adopting security practices in agile software development is focusing on functionality rather than security.

Experience shows that product owners should not only be measured by the fast delivery of functional, user-centric requirements and meeting business goals, but their goals should also include non-functional requirements like security. Another area for improvement is the paucity of documentation and support for security practices. At the beginning of the transformation, there was a lack of guidance on how to integrate security practices into agile development. The absence of a clearly responsible entity in the landscape of self-organizing teams further increased this challenge. Introducing the security guild tackled this challenge and succeeded in incorporating security into the agile organization. This identified obstacle matches the finding of Oyetoyan et al. [41] that an enabling environment is required to implement security approaches in agile software development effectively. How an organization drives the adoption initiative can either be a driving or a hindering factor for introducing agile security approaches. One interviewee is sure that "if [one] want[s] to introduce approaches within such a large corporation, they have to be delegated from the top." This involves the commitment of top-level management. On the other hand, one interviewee pointed out that the top-down introduction of approaches must be done "with a lot of tact" so as not to cause resentment among the predominantly self-organized teams. Most importantly, bottom-up participation should be encouraged, i.e., agile teams should be involved in evaluating and selecting security approaches. The interviewed experts also explained that the adoption process needs to rely on known wordings and examples to make it easy for agile team members.

5 EVALUATION OF AGILE SECURITY APPROACHES

This section presents the evaluation results of 14 preselected agile security approaches. We first present the assessment of four role and knowledge approaches based on the quantitative results of the evaluation questionnaire. Figure 1 shows an excerpt from the survey results. Secondly, we summarize the findings of the interviews we primarily used to evaluate the 10 methodology approaches.

5.1 Evaluation of role and knowledge approaches

5.1.1 Security knowledge exchange. The security guild is central to adopting security approaches in the case company. 32 of 53 respondents assess the guild as very valuable or valuable for their agile tribe and teams. Only two participants tend to see no value in the guild's activities. About 72% of the respondents think the security guild will sustainably contribute to an appropriate and policy-compliant security level in the agile software development teams. 13% do not believe in a lasting impact of the guild. 40 of 53 question respondents believe that cross-team knowledge exchange has increased through the security guild offerings. A product owner states that there is "more transparency and exchange among each other." A Scrum master values that cross-team problems are now

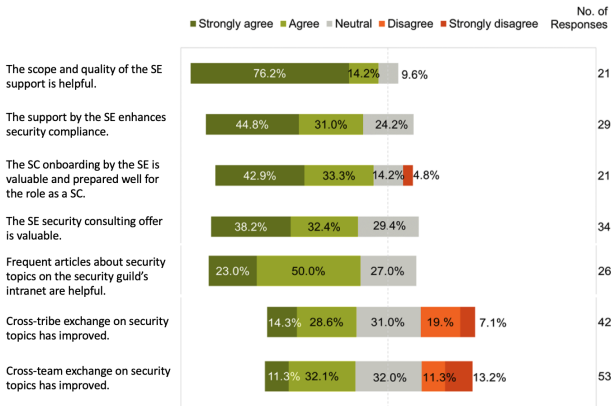

Figure 1: Excerpt from the survey results on the evaluation of role and knowledge approaches.

much more visible, and solutions can be found together. Nevertheless, the survey results also show potential for improvement. 26% of the respondents do not see an improved cross-tribe exchange on security yet, and almost 25% do not yet see significant progress in cross-team exchange. Furthermore, only half of the survey participants know all the security guild offerings in the enterprise social network. 65% of them use or read content from the security guild monthly or more frequently, and a little more than 60% of these participants consider the frequent articles about security topics as very or rather useful. However, some SCs express the preference of an email newsletter rather than having to pull information themselves.

5.1.2 Team-external security role. 70% of the respondents rate the support by SEs as very or rather valuable. Participants especially value having a clearly defined contact person for security and the high security competency. The 21 SCs participating in the survey assessed two additional aspects concerning the SEs. Around 76% of all participating SCs specify that the onboarding and mentoring by their SE prepared them very or fairly well for their role. One potential improvement mentioned in the survey responses is that the SC role could be more clearly defined. More than three-quarters of all respondents indicate that SE support helps them to better adhere to the security guidelines during the implementation of their applications. According to the responses, SEs also help reduce common problems, e.g., generic wording of security policies that leaves open questions during implementation or complex security requirements spreading out over multiple documents.

5.1.3 Team-internal security expert. At the time of the survey, the case company was still establishing the new role of a SC. Nevertheless, 21 survey respondents were already in the role of a SC. The survey reveals that the SC role is already perceived as valuable and that most SCs are interested in contributing to the security guild. To be able to support their teams with security concerns and fulfill the responsibility of spreading security know-how, they need sufficient training. They especially value training courses and examples with practical relevance, such as the use of security testing tools or secure coding workshops. The responses indicate that some SCs

still need to gain more experience until they feel comfortable in their role. For this purpose, they also asked for more specialized training, e.g., security risk assessments.

5.1.4 Security training. Almost 60% rate the guild's training offering as very valuable or rather valuable. They appreciate that the training is offered in different formats, such as video or text, since it "helps different learning types to understand the concepts." The results show that the interest of agile developers in security training is high. 95% of the 22 agile team members aware of the guild's training offering have increased or at least partially increased their personal expertise on security topics through this offering. In addition, more than 80% plan to continue using it in the future. 23 out of 27 individuals who evaluated the security guild's monthly information and training meetings consider them valuable or partly valuable.

5.2 Evaluation of methodology approaches

Table 3 presents the aggregated results of all detailed expert evaluations per approach and evaluation criterion. Interviewees rated each approach for each criterion as positive, neutral, or negative. The resulting scores are the arithmetic means of all individual ratings of the interviewed experts. This scores the criteria on a scale of minus one to one, with one being the most positive score and minus one being the most negative.

Table 3: Evaluation results of agile security methodology approaches per evaluation criteria

Agile criteria	Security stories	Criteria in DoD	Tagging system	Attack tree/ CM graph	Threat Poker	pair pen. Testing	Peer code Reviews
Iterative	1	1	1	1	1	1	1
Adaptable	0.7	-0.5	1	0.5	1	0	0.8
Interactive	0.8	-0.5	0	1	1	0.8	0.8
Lean	0.3	1	0	-1	0	0.5	0.4
Simple	0.3	1	1	-1	1	-0.3	0.4
Motivating	0.2	-0.5	-0.3	0.5	1	1	0.6
Large-scale criteria							
Autonomy	0.3	0	0.7	-1	0	-0.3	0.4
Collaboration	0.7	0.5	1	0.5	1	0.8	0.6
Knowledge Exchange	0.8	-0.5	0	0.5	1	0.3	0.6
Technical Consistency	0.5	1	0.3	0	0	0.8	0.2
Regulatory criteria							
Traceability	0.2	0.5	1	-0.5	0.5	-0.8	-0.2
Compliance	1	0.5	0.7	0.5	1	0.5	0.2

5.2.1 Security stories & misuse/abuse(r) cases. Eight of the 10 interviewed professionals provided their opinion on security stories in conjunction with misuse/abuse(r) cases. Six consider them one of the most relevant and suitable approaches for agile development environments. The reasons are the approach's practical nature and the seamless integration into agile methods because of the similarity to user stories. Other frequently mentioned advantages are increased transparency of security threats and enhanced security awareness of the agile teams. Several experts agree that through the early consideration of security, the approach leads to a leaner software development process. Mentioned challenges are that permanently tracking misuse cases is time-consuming and requires a security expert with enough experience in attack scenarios and vulnerability detection. Four interviewees consider an increase in autonomy possible. The approach enables the teams to identify security risks and proactively improve the security level of their product. Furthermore, several interviewees think that security stories improve cross-team collaboration. They suggested applying the approach in planning meetings on a tribe level. Finally, all interviewees assessing the approach in detail indicated that it facilitates compliance with guidelines.

5.2.2 Security backlog. Nine of the 10 interviewed experts discussed the approach of a security backlog. Four interviewees are critical of a separate security backlog. An additional security backlog parallel to the product backlog causes extra effort and might be ignored. An isolated security backlog might also intensify the prioritization conflict between functional and security requirements. Nevertheless, several interview partners find the explicit gathering of security stories in a separate backlog useful, e.g., it might make sense to collect regulatory security stories in a security backlog and move them to the product backlog, if required for the specific system.

5.2.3 Security Criteria in Definition of Done (DoD). Five participants mentioned security criteria in the DoD. The named advantages are that security is integrated as a fixed component in the DoD from the outset and that it increases transparency. A challenge is choosing an appropriate concreteness and number of security criteria. One expert remarks that "if I formulate [the DoD] very roughly, it is so unspecific that it is useless, and if it is very detailed, there are too many checkpoints [...] for each [user story]." Another identified challenge is the difficulty for product owners to control the adherence to the DoD since they often do not have such a profound expertise in secure software development to be able to verify those criteria. One expert considers DoD criteria as not adaptable and easily customizable because teams typically set them once and do not keep changing them.

5.2.4 Agile risk analysis: security tagging system. The security tagging system presented to the interviewees combines agile risk analysis [15, 34] and the security story variant of Pohl and Hof [43]. It includes identifying and marking security-relevant user stories, conducting a risk analysis, and prioritizing according to risk levels. All seven interviewees who mentioned this method rated it mostly positively. The main benefits of the security tagging system are the overview and transparency of security-relevant topics for SEs and SCs, the ease of integrating it into the daily work of agile teams, and the increase in security awareness. One expert summarized that the approach "creates awareness and starts directly where the teams operate." As teams perform the identification of security-relevance, risk assessment, and labeling with user stories, they have flexibility in moving them to subsequent iterations or adapt them in case of requirement changes. The experts agree that the security tagging system is easy to understand and "one of the most intuitive approaches." One expert proposed to provide checklists to teams that support the security-relevance and risk assessment. The approach "can make teams more autonomous in the sense that it is easier for them to prove that they have really paid attention to security on an ongoing basis [...]."

5.2.5 Agile risk analysis: protection poker. None of the interviewed professionals consider protection poker as one of the most relevant or suitable approaches for regulated LSAD environments. However,

they expressed that this collaborative game might be more motivating and fun for agile teams than the security tagging system. An expert explained that protection poker is "perceived more as an activity the teams already know and less as an extra documentation effort." In addition, an interviewee explained that it "would be a promising approach because it involves the whole team and ultimately creates awareness for security."

5.2.6 Agile threat analysis & modelling: attack trees & countermeasure graphs. Four interviewed experts view attack trees and countermeasure graphs as potentially partly suitable and relevant for the agile development environment. They outlined the advantages of a positive impact on application security and awareness as well as increased transparency of threats and chains of security attacks. Mentioned criticism includes that attack trees are "too abstract, academic and far away from development practices." Additionally, the approach becomes very laborious for dynamic components that change often. One expert proposed to execute the approach for critical applications during a separate event or workshop with the support of security experts.

5.2.7 Agile threat analysis & modelling: threat poker. Among the benefits of the approach mentioned by five experts are the "fun factor" and a potential rise in the agile teams' vigilance towards security topics and threats. It trains the team in assessing the probability of occurrence, potential damage, and elimination of security risks. Another advantage is that it results in a cost and effort estimation for implementing countermeasures. It is also adaptive because if a threat changes or new ones arise, it can be played independently of any specific sprints. The experts acknowledge the game to be interactive, playful, and people-based. Regarding challenges, they agree that the approach is time-consuming in the short term. However, threat poker "will simplify things in the long run and probably also prevent [penetration test] findings." An enhancement in the traceability of security requirements is possible if the assessments of threat scenarios and countermeasures are documented during the game.

5.2.8 Security meetings & sprints. Seven experts addressed security meetings, sprints, and spikes during the interviews. Two interviewees noted that extra security meetings take up more time, and developers, in particular, aim to shorten meeting times and may not welcome additional effort. However, four experts believe that security sprints would increase security in agile software development. They propose using them to resolve known vulnerabilities and tackle security issues that cannot be dealt with in usual sprints or day-to-day business. However, some believe it is unrealistic that product owners or sponsors approve the idea of security sprints because it would pause feature development. Hence, interviewees consider security spikes more achievable than security sprints.

5.2.9 Pair penetration testing. Interviewees rated pair penetration testing as "very useful for the teams" and highlighted the potential for significant knowledge acquisition as a key advantage. Agile developers build a better understanding of the thought processes of a penetration tester and how they perform a test. Two interviewed experts believe that this improved understanding leads to more motivation among the agile team members. One interviewee summarized that "the key point [of the approach] is to really see how security works. It is a very, very exciting topic and a useful tool to motivate people to look at security issues." Another interviewee described that it "can be exciting and interesting when trying to break into [one's] application." Nevertheless, the interviewed experts mentioned two main drawbacks of pair penetration testing. First, penetration tests are often performed by external providers, and it is unclear if they are willing to pair during the tests in a way that adds value and has a learning effect. Secondly, the idle time that might occur during testing is less valuable because penetration testers often have to experiment extensively before they discover a vulnerability.

5.2.10 Peer security code review. All but one of the experts assessed peer security code reviews during the interviews. Seven find the approach suitable and relevant for regulated LSAD environments. Mentioned reasons are that code review is an activity most developers are familiar with and that it directly addresses the actual implementation. Two interviewees value the security knowledge building and social aspect of the approach. Several experts support the SC as the leading peer partner for other developers, at least initially, until other team members build enough competency to conduct code reviews focusing on security. Five interviewed professionals evaluated peer security code reviews as being highly compatible with iterative development. In addition, they consider them to be adaptive, interactive, and fostering team autonomy, especially when performed by team members without an external security expert. However, experts raised concerns that the effectiveness of the approach relies on the feedback culture and that it can impede development due to the additional resource required for the review. Furthermore, they argue that identifying and fixing problems through the review saves time in the long run. One expert highlights that it is not short-term speed that is optimized, but quality. Even if some experts agree that the approach is easy to understand, most note that successful reviews need a certain amount of security experience, practice, and supporting checklists. Three experts strongly believe that peer security code reviews are highly motivating due to the interaction with the reviewer and the collaborative effort dedicated to addressing the issues at hand. Using peers outside the team can also foster cross-team collaboration and knowledge sharing.

6 DISCUSSION

We address our research questions by discussing the key findings and outlining the limitations.

6.1 Key findings

Our examination of agile security approaches in the case company shows the importance of security role and knowledge approaches in LSAD. Although relevant academic publications describe a higher number of methodology approaches, they acknowledge the importance of roles and knowledge approaches in LSAD settings. Other researchers suggest combining both categories of approaches for a successful integration of security in LSAD. This is also evident in the examined organization. The company first implemented security roles during the scaling of the agile structure to facilitate building and exchanging security knowledge. Afterward, they addressed adopting security methodology approaches within the agile teams.

Based on our evaluation results of the 10 methodology approaches identified in relevant literature and assessed by 10 agile security experts, the ratings are most favorable for the security tagging system, peer security code reviews, security stories in conjunction with misuse cases, and threat poker. Considering the case study organization's context, threat poker is potentially most suitable for small-scale agile settings, while security stories are potentially most appropriate for LSAD. The security tagging system is possibly best suited for regulated environments, as shown in table 3. Of the four agile security role and knowledge approaches evaluated with the survey, the team-external security role is the most successfully adopted and beneficial. The SC role is promising, but more experience is required to leverage the added value fully. We expected this outcome since the role was still relatively new at the time of our case study. The survey results show that cross-team and cross-tribe security knowledge exchange is still a challenge in the LSAD environment of the case company, even if a majority of respondents already see a clear improvement through the introduced security guild community and the security roles. We also observe that it is essential to communicate all security support offerings clearly since only half of the survey participants know all the security guild offerings in the case company. The evaluation criteria and identified advantages and drawbacks could also reasonably be applied in other companies and industries. Even if specific details might need to be adapted, our work provides a valuable basis for preselecting and evaluating agile security approaches.

Furthermore, our analysis reveals 11 driving and hindering factors for adopting security approaches in LSAD. Although scientific articles rarely discuss drivers and obstacles in the adoption process, six of the 11 identified factors are in line with findings from the literature. Security awareness and knowledge, investment in resources, and intrinsic motivation of developers are among the key drivers for adopting agile security approaches. Among the main obstacles is the need for more security awareness and resources. Reasons for this shortcoming are the lack of experience of agile teams with security topics and the low familiarity with agile security approaches. In addition, as expected, the investment of resources such as time and budget by key stakeholders like customers and agile project sponsors is critical to the adoption process.

6.2 Limitations

Although we used a thorough methodology, this research work has several limitations. The conducted systematic literature review aiming to provide a basis for identifying agile security approaches in the case company cannot guarantee completeness, since some relevant publications may be missing [54]. For our case study, we consider reliability, construct validity, and external validity [33, 55]. To ensure reliability, we documented the purpose, questions, and theoretical framework of the study, as well as the data collection methods and results, similar to the case study protocol proposed by Yin [55]. Researchers might also introduce bias during interview transcriptions and data interpretation. To minimize this threat, we followed established guidelines for interview analysis, as described in 3.2.3.

To address the potentially limited construct validity, we clarified any ambiguity or questions the interviewees might have directly during the interviews. We also repeatedly summarized the key statements of the interview partners to avoid misunderstandings. For the survey, we aimed to ensure the comprehensibility of the questionnaire through iterative feedback loops with pilot groups. Since we conducted a single-case study, a primary concern is external validity. To achieve a degree of generalizability, we did not solely focus on the specifics of the case company. Instead, we have developed a generic set of evaluation criteria that enables the assessment of security approaches in agile, LSAD, and regulated environments. The 10 interviewed experts all had prior experience outside the case company, which they brought into the interviews. In addition, we found similarities between our findings and empirical research in academic literature, which indicates further potential transferability. Therefore, the insights of this work should be valuable for other organizations and researchers. Due to our study's limited time frame and the industry experts' capacity, we could only assess some of the 27 identified approaches. Therefore, we preselected based on the criteria described in 3.2.4.

7 CONCLUSION AND FUTURE WORK

As security challenges rise and agile software development at scale becomes increasingly ubiquitous, integrating security into LSAD proves to be crucial. With our case study, we contribute to the empirical evidence by exploring the adoption of security practices in a LSAD environment. Of the 27 approaches we identified in a preliminary literature review, we found the case company to use 11. We identified those mainly by applying observation, unstructured interviews, and document analysis. These include four role and knowledge approaches: security training, security knowledge exchange, security experts in agile teams, and a team-external security role. Furthermore, they include methodology approaches such as a tagging system for security stories, security criteria in the DoD, security documentation artifacts, security testing, and security code review. We evaluated 14 selected approaches through interviews with 10 experts and a survey with 62 participants. We also discovered 11 drivers and obstacles for adopting agile security activities in the process.

Our evaluation results reveal the potential suitability of certain activities, their advantages, and drawbacks. They can be used as valuable input for other organizations' decision-making and adoption process or as a basis for further research. To improve the generalization of the results and complement empirical evidence, further studies could examine the adoption of agile security approaches in other companies and industries. Focusing on security activities suitable for regulated LSAD environments would be especially valuable because existing research in these settings is scarce. In addition, further exploring driving and hindering factors in the adoption process as well as evaluation criteria for agile security approaches could be useful for both researchers and practitioners. To validate the acceptance and usefulness of the approaches rated as potentially most suitable, experiments or a field study over an extended period would be an intriguing next research opportunity.

ACKNOWLEDGMENTS

This work has been supported by the German Federal Ministry of Education and Research (BMBF) Software Campus grant 01IS17049.

REFERENCES

[1] Julia H Allen, Sean Barnum, Robert J Ellison, Gary McGraw, and Nancy R Mead. 2008. *Software Security Engineering: A Guide for Project Managers*. Vol. 1. Addison-Wesley Professional, Boston.

[2] Wasim Alsaqaf, Maya Daneva, and Roel Wieringa. 2019. Quality requirements challenges in the context of large-scale distributed agile: An empirical study. *Information and Software Technology* 110 (2019), 39–55. https://doi.org/10.1016/j.infsof.2019.01.009

[3] Adila Firdaus Arbain, Imran Ghani, and Seung Ryul Jeong. 2014. A Systematic Literature Review on Secure Software Development using Feature Driven Development (FDD) Agile Model. *Journal of Korean Society for Internet Information* 15, 1 (2014), 13–27. https://doi.org/10.7472/jksii.2014.15.1.13

[4] Dejan Baca, Martin Boldt, Bengt Carlsson, and Andreas Jacobsson. 2015. A Novel Security-Enhanced Agile Software Development Process Applied in an Industrial Setting. In *10th International Conference on Availability, Reliability and Security*. IEEE Computer Society, NW Washington, DC, USA, 11–19. https://doi.org/10.1109/ARES.2015.45

[5] Dayanne Araujo Barbosa and Suzana Sampaio. 2015. Guide to the Support for the Enhancement of Security Measures in Agile Projects. In *2015 6th Brazilian Workshop on Agile Methods (WBMA)*. IEEE, 25–31. https://doi.org/10.1109/WBMA.2015.9

[6] Steffen Bartsch. 2011. Practitioners' perspectives on security in agile development. *IEEE Sixth International Conference on Availability, Reliability and Security* (2011), 479–484. https://doi.org/10.1109/ARES.2011.82

[7] J. M. Bass. 2016. Artefacts and Agile Method Tailoring in Large-Scale Offshore Software Development Programmes. *Information and Software Technology* 75 (2016), 1–16. https://doi.org/10.1016/j.infsof.2016.03.001

[8] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas. 2001. Manifesto for Agile Software Development. http://agilemanifesto.org/

[9] Laura Bell, Michael Brunton-Spall, Rich Smith, and Jim Bird. 2017. *Agile Application Security: Enabling Security in a Continuous Delivery Pipeline*. O'Reilly Media Inc.

[10] Carlos Magnum M. Bezerra, Suzana C. B. Sampaio, and Marcelo L. M. Marinho. 2020. Secure Agile Software Development: Policies and Practices for Agile Teams. In *Quality of Information and Communications Technology*, Martin Shepperd, Fernando Brito e Abreu, Alberto Da Rodrigues Silva, and Ricardo Pérez-Castillo (Eds.). Communications in Computer and Information Science, Vol. 1266. Springer International Publishing, Cham, 343–357. https://doi.org/10.1007/978-3-030-58793-2_28

[11] Konstantin Beznosov and Philippe Kruchten. 2005. Towards Agile Security Assurance. *Proceedings of the 2004 Workshop on New Security Paradigms* (2005), 47–54.

[12] Dave Bishop, Rowland, and Pam. 2019. AGILE AND SECURE SOFTWARE DEVELOPMENT: AN UNFINISHED STORY. *Issues in Information Systems* 20, 1 (2019), 144–156.

[13] Alexander Bogner, Beate Littig, and Wolfgang Menz. 2014. *Interviews mit Experten: eine praxisorientierte Einführung*. Springer Fachmedien, Wiesbaden.

[14] Jürgen Bortz and Nicola Döring. 1995. *Forschungsmethoden und Evaluation*. Springer, Berlin, Heidelberg.

[15] Gustav Boström, Jaana Wäyrynen, Marine Bodén, Konstantin Beznosov, and Philippe Kruchten. 2006. Extending XP Practices to Support Security Requirements Engineering. *Proceedings of the 2006 international workshop on Software engineering for secure systems* (2006), 11–18.

[16] Maya Daneva and Chong Wang. 2018. Security Requirements Engineering in the Agile Era: How Does it Work in Practice?. In *2018 IEEE 1st International Workshop on Quality Requirements in Agile Projects (QuaRAP)*. IEEE, 10–13. https://doi.org/10.1109/QuaRAP.2018.00008

[17] Kim Dikert, Maria Paasivaara, and Casper Lassenius. 2016. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software* 119 (2016), 87–108. https://doi.org/10.1016/j.jss.2016.06.013

[18] Torgeir Dingsøyr, Tor Erlend Fægri, and Juha Itkonen. 2014. What Is Large in Large-Scale? A Taxonomy of Scale for Agile Software Development. In *Product-Focused Software Process Improvement*, Andreas Jedlitschka, Pasi Kuvaja, Marco Kuhrmann, Tomi Männistö, Jürgen Münch, and Mikko Raatikainen (Eds.). Lecture notes in computer science, Vol. 8892. Springer International Publishing, Cham, 273–276. https://doi.org/10.1007/978-3-319-13835-0_20

[19] Torgeir Dingsøyr, Nils Brede Moe, Tor Erlend Fægri, and Eva Amdahl Seim. 2018. Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering* 23, 1 (2018), 490–520. https://doi.org/10.1007/s10664-017-9524-2

[20] Torgeir Dingsøyr, Sridhar Nerur, VenuGopal Balijepally, and Nils Brede Moe. 2012. A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software* 85, 6 (2012), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033

[21] Kathleen M Eisenhardt. 1989. Building theories from case study research. *Academy of management review* 14, 4 (1989), 532–550.

[22] Brian Fitzgerald, Klaas-Jan Stol, Ryan O'Sullivan, and Donal O'Brien. 2013. Scaling Agile Methods to Regulated Environments: An Industry Case Study. In *International Conference on Software Engineering (ICSE)*. Vol. 35. 863–872.

[23] European Union Agency for Cyber Security. 2023. ENISA Threat Landscape 2022. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2022

[24] Floyd J Fowler Jr. 2009. *Survey research methods* (4 ed.). SAGE publications.

[25] Imran Ghani, Zulkarnain Azham, and Seung R. Jeong. 2014. Integrating software security into agile-Scrum method. *KSII Transactions on Internet and Information Systems (TIIS)* 8, 2 (2014), 646–663. https://doi.org/10.1109/MySEC.2011.6140708

[26] Lise Tordrup Heeager and Peter Axel Nielsen. 2018. A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology* 103 (2018), 22–39. https://doi.org/10.1016/j.infsof.2018.06.004

[27] Rashina Hoda, Norsaremah Salleh, and John Grundy. 2018. The Rise and Evolution of Agile Software Development. *IEEE Software* 35, 5 (2018), 58–63. https://doi.org/10.1109/MS.2018.290111318

[28] International Organization for Standardization. 2018. ISO/IEC 27000:2018(en): Information technology - Security techniques - Information security management systems - Overview and vocabulary.

[29] Gibrail Islam and Tim Storer. 2020. A case study of agile software development for safety-Critical systems projects. *Reliability Engineering & System Safety* 200 (2020), 106954. https://doi.org/10.1016/j.ress.2020.106954

[30] Hossein Keramati and Seyed-Hassan Mirian-Hosseinabadi. 2008. Integrating Software Development Security Activities with Agile Methodologies. *IEEE/ACS International Conference on Computer Systems and Applications* (2008), 749–754.

[31] Raja Khaim, Saba Naz, Fakhar Abbas, Naila Iqbal, and Memoona Hamayun. 2016. A Review of Security Integration Technique in Agile Software Development. *International Journal of Software Engineering & Applications* 7, 3 (2016), 49–68. https://doi.org/10.5121/ijsea.2016.7304

[32] Henrik Kniberg and Anders Ivarsson. 2012. Scaling agile@ spotify. *online], UCVOF, ucvox. files. wordpress. com/2012/11/113617905-scaling-Agile-spotify-11. pdf* (2012).

[33] Steinar Kvale. 2007. *Doing interviews* (1 ed.). SAGE publications.

[34] Patrik Maier, Zhendong Ma, and Roderick Bloem. 2017. Towards a Secure SCRUM Process for Agile Web Application Development. In *Proceedings of the 12th International Conference on Availability, Reliability and Security*. ACM, New York, NY, USA, 1–8. https://doi.org/10.1145/3098954.3103171

[35] Philipp Mayring. 2015. *Qualitative Inhaltsanalyse: Grundlagen und Techniken*. Beltz.

[36] Gary McGraw. 2006. *Software Security: Building Security In*. Addison-Wesley.

[37] Fabiola Moyón, Pamela Almeida, Daniel Riofrio, Daniel Mendez, and Marcos Kalinowski. 2020. Security Compliance in Agile Software Development: A Systematic Mapping Study. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 413–420. https://doi.org/10.1109/SEAA51224.2020.00073

[38] Sascha Nägele, Jan-Philipp Watzelt, and Florian Matthes. 2022. Investigating the Current State of Security in Large-Scale Agile Development. In *Agile Processes in Software Engineering and Extreme Programming: 23rd International Conference on Agile Software Development, XP 2022, Copenhagen, Denmark, June 13–17, 2022, Proceedings*. Springer, 203–219.

[39] Nathan Newton, Craig Anslow, and Andreas Drechsler. 2019. Information Security in Agile Software Development Project: A Critical Success Factor Perspective. *Proceedings of the 27th European Conference on Information Systems (ECIS)* (2019).

[40] Torstein Nicolaysen, Richard Sassoon, Maria B. Line, and Martin Gilje Jaatun. 2010. Agile Software Development. *International Journal of Secure Software Engineering* 1, 3 (2010), 71–85. https://doi.org/10.4018/jsse.2010070105

[41] Tosin Daniel Oyetoyan, Daniela Soares Cruzes, and Martin Gilje Jaatun. 2016. An Empirical Study on the Relationship between Software Security Skills, Usage and Training Needs in Agile Settings. In *2016 11th International Conference on Availability, Reliability and Security (ARES)*. IEEE, 548–555. https://doi.org/10.1109/ARES.2016.103

[42] Kai Petersen and Claes Wohlin. 2010. The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering* 15, 6 (2010), 654–693. https://doi.org/10.1007/s10664-010-9136-6

[43] Christoph Pohl and Hans-Joachim Hof. 2015. Secure Scrum: Development of Secure Software with Scrum. http://arxiv.org/pdf/1507.02992v1

[44] Check Point Research. 2023. Check Point Research Reports a 38% Increase in 2022 Global Cyberattacks. https://blog.checkpoint.com/2023/01/05/38-increase-in-2022-global-cyberattacks/

[45] Kalle Rindell, Sami Hyrynsalmi, and Ville Leppänen. 2015. Securing Scrum for VAHTI. *14th Symposium on Programming Languages and Software Tools* (2015). https://doi.org/10.13140/RG.2.1.4660.2964

[46] Kalle Rindell, Sami Hyrynsalmi, and Ville Leppänen. 2017. Busting a Myth: Review of agile security engineering methods. *Proceedings of the 12th International*

Conference on Availability, Reliability and Security (2017), 1–10. https://doi.org/10.1145/3098954.3103170

[47] Kalle Rindell, Jukka Ruohonen, Johannes Holvitie, Sami Hyrynsalmi, and Ville Leppänen. 2021. Security in agile software development: A practitioner survey. *Information and Software Technology* 131 (2021), 106488. https://doi.org/10.1016/j.infsof.2020.106488

[48] Kalle Rindell, Jukka Ruohonen, and Sami Hyrynsalmi. 2018. Surveying Secure Software Development Practices in Finland. *Proceedings of the 13th International Conference on Availability, Reliability and Security* (2018), 1–7. https://doi.org/10.1145/3230833.3233274

[49] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* 14, 2 (2009), 131–164. https://doi.org/10.1007/s10664-008-9102-8

[50] M. Siponen, R. Baskerville, and T. Kuivalainen. 2005. Integrating Security into Agile Development Methods. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*. 185a–185a. https://doi.org/10.1109/HICSS.2005.329

[51] Jan-Philipp Steghöfer, Eric Knauss, Jennifer Horkoff, and Rebekka Wohlrab. 2019. Challenges of Scaled Agile for Safety-Critical Systems. In *International Conference on Product-Focused Software Process Improvement*, Vol. 11915. Springer, Cham, 350–366. https://doi.org/10.1007/978-3-030-35333-9_26

[52] TemboSocial. 2023. TemboSocial: Software for Employee Recognition and Employee Feedback. https://www.tembosocial.com/

[53] VersionOne. 2022. 16th Annual State of Agile Report. https://stateofagile.com/

[54] Jane Webster and Richard T. Watson. 2002. Analyzing the Past to Prepare for the Future: Writing a literature Review. *MIS Quarterly* 26, 2 (2002), xiii – xxiii.

[55] Robert K. Yin. 2003. *Case Study Research: Design and Methods* (3th edition ed.). Sage, Thousand Oaks.

[56] Ömer Uludağ, Pascal Philipp, Abheeshta Putta, Maria Paasivaara, Casper Lassenius, and Florian Matthes. 2022. Revealing the state of the art of large-scale agile development research: A systematic mapping study. *Journal of Systems and Software* 194 (2022), 111473. https://doi.org/10.1016/j.jss.2022.111473

Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development

Sascha Nägele, Nathalie Schenk, Nico Fechtner and Florian Matthes
School of Computation, Information and Technology, Technical University of Munich, Germany
{*sascha.naegele, nathalie.schenk, nico.fechtner, matthes*}*@tum.de*

Keywords: Large-Scale Agile Development, Security, Governance, Compliance.

Abstract: Companies are increasingly adopting agile methods at scale, revealing a challenge in balancing team autonomy and organizational control. To address this challenge, we propose an adaptive approach for security governance in large-scale agile software development, based on design science research and expert interviews. In total, we carried out 28 interviews with 18 experts from 15 companies. Our resulting approach includes a generic organizational setup of security-related roles, a team autonomy assessment model, and an adaptive collaboration model. The model assigns activities to roles and determines their frequency based on team autonomy, balancing the autonomy-control tension while ensuring compliance. Although framework-agnostic, we applied our approach to existing scaling agile frameworks to demonstrate its applicability. Our evaluation indicates that the approach addresses a significant problem area and provides valuable guidance for incorporating security into scaled agile environments. While the primary focus is on security governance, our insights may be transferable to other cross-cutting concerns.

1 INTRODUCTION

Driven by the success of agile software development (ASD) methodologies, the application of agile approaches at scale has seen a marked increase (Uludağ et al., 2022). Scaling such development efforts is complex, particularly when there are stringent quality requirements to be met (Kalenda et al., 2018). This complexity raises the issue of balancing team autonomy and decentralized decision-making against the necessity for organizational control and alignment in large-scale agile development (LSAD) (Nägele et al., 2022).

Concurrently, security, a paramount information systems and software quality requirement, is growing in significance due to increasing threats and resulting regulatory requirements (Tayaksi et al., 2022; Moyón et al., 2021). This trend intensifies the tension between autonomy and control, elevating security governance and compliance to a critical concern in LSAD (Moyón et al., 2021).

Existing research indicates that established scaling agile frameworks neither adequately incorporate security activities nor provide sufficient guidance on security and security compliance (Edison et al., 2021; Nägele et al., 2023). For instance, Dännart et al. (2019) observe that security is frequently treated as an isolated concern, and there is a deficiency in "knowing where to conduct which security activity in a large scale agile process" (p. 531). Moyón et al. (2021) suggest that adapting scaling frameworks, e.g., by introducing additional roles, may be beneficial. In addition, based on a systematic literature review and expert interview study, Nägele, Schenk, and Matthes (2023) propose employing influencing factors such as product risk and team maturity to tailor governance and control mechanisms, thereby alleviating the identified tension and enabling increased autonomy and responsibility for more proficient teams. Despite these advances, guidance on how to establish a suitable balance between autonomy and control is lacking.

To fill this research gap, we propose an adaptive approach that aims to enable agile team autonomy while maintaining effective development control without imposing unnecessary burdens. Our primary goal is to establish a balance between agility and (security) governance, resulting in a clear secure software development process that is systematic, transparent, and auditable, while ensuring compatibility with agile environments at scale. We aim to achieve this integration by emphasizing collaboration, continuous improvement, and team autonomy.

Our research question (RQ) is: *How can secu-*

Nägele, S., Schenk, N., Fechtner, N. and Matthes, F.
Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development.

In *Proceedings of the 26th International Conference on Enterprise Information Systems (ICEIS 2024) - Volume 2*, pages 17-28
ISBN: 978-989-758-692-7; ISSN: 2184-4992

rity governance be integrated into LSAD while ensuring adaptability to achieve a suitable equilibrium between autonomy and control?

In the pursuit of answering our RQ, we developed an adaptive approach using design science research (DSR) and expert interviews.

The approach systematically adapts autonomy by considering key influencing factors such as product risk and the capability of agile development teams, granting higher autonomy to more mature teams. It also proposes an organizational setup to foster security collaboration in agile environments at scale.

The paper is structured as follows. Section 2 covers background and related work. Section 3 explains our research method. Section 4 introduces our adaptive approach for security governance in LSAD, evaluated in Section 5. Sections 6 and 7 discuss our findings and conclude our research.

2 BACKGROUND AND RELATED WORK

The following section summarizes the most important definitions as well as the existing challenges and solutions that are relevant to our RQ.

2.1 Security Governance and Compliance

When using the term *governance*, we mainly refer to the domain of *information security governance*, which encompasses the organizational structure and processes to ensure information security. Information security aims to safeguard "the integrity of information, continuity of services and protection of information assets" assets" (Williams, 2001, p. 64). The closely related term *security compliance* can be defined as "the state of conformance with [...] imposed functional security requirements and of providing evidence (assurance) thereof" thereof" (Julisch, 2008, p. 72). The origin of these requirements could be either internal or external (such as required by regulatory bodies, industry standards, or clients).

2.2 Challenges of Security in ASD

To fulfill these internal and external requirements, organizations frequently resort to non-agile security development life cycles, incorporating linear governance and compliance processes within ASD methodologies (Rindell et al., 2018).

This dichotomy presents significant challenges, as widely acknowledged in the current literature. For instance, a systematic mapping study by Moyon et al. (2020) reveals that ensuring security compliance often poses challenges for organizations employing ASD methods, as the required effort grows with shorter development and delivery cycle times. Activities such as integrating security experts and assessors or documenting security compliance evidence can present major hurdles.

Furthermore, Bartsch (2011) identified three main types of challenges for integrating security in agile development, based on literature: challenges related to process aspects, communication and interaction, and trust in individuals and teams.

Trust is closely related to the concept of autonomous teams in ASD. When speaking of *team autonomy*, we refer to "self-organizing teams" or "self-managing teams", as they are synonymously described and defined by researchers in the area of autonomous agile teams (Stray et al., 2018).

2.3 Challenges of Security in LSAD

The tension between agility and security is amplified when trying to scale agile processes while maintaining security compliance without compromising flexibility and speed (Moyón et al., 2021). In regards to scaling agile processes, Dikert et al. (2016) reviewed multiple definitions of LSAD and propose to define *large-scale agile development* as a "software organization with 50 or more people or at least six teams" (Dikert et al., 2016, p. 88).

While the majority of the literature focuses on the general friction between agile methodologies and security, a few authors already investigated security challenges particularly in LSAD.

Nägele, Schenk, and Matthes (2023) identified 15 challenges based on a systematic literature review and interview study, partly based on van der Heijden et al. (2018) who previously identified security challenges specific to LSAD environments. One of the key challenges is the prevalent goal of agile teams to pursue team autonomy, while a certain level of control is often still required, especially in regulated environments (Nägele et al., 2023).

Additionally, Edison et al. (2021) identified several security-related challenges when implementing scaling agile frameworks in practice, such as deficient security awareness and the absence of proactive security activities.

We address the identified challenges in the design of our solution artifact as described in Section 4.1.

2.4 Solution Approaches

In addition to addressing known challenges, we draw from existing solution approaches and success factors already described in the existing literature.

Nägele, Schenk, and Matthes (2023) propose five factors to balance team autonomy and organizational control through security governance based on a systematic literature review and interview study. Newton et al. (2019) propose propose twelve critical success factors and related practices to integrate information security within ASD. These include achieving security awareness, security training, early security testing, dedicated security activities, and a straightforward release process considering security. Typical examples of security activities suitable for LSAD environments are threat modeling, security self-assessments, code reviews, or penetration tests (Nägele et al., 2022).

Bartsch (2011) asserts that integrating security and agility could be achieved by promoting a unified understanding of roles and effectively assimilating assurance and documentation activities into the agile process. Bell et al. (2017) conclude that it is possible to develop secure software with agile approaches if, on the one hand, agile teams integrate security activities and are responsible for developing secure software and, on the other hand, receive enough guidance from security experts. It may be helpful for teams to get involved with security if one team member (a designated security expert or a developer) takes over a security role within the team (Bell et al., 2017).

2.5 Similar Approaches

In the course of our research, two publications emerged with partly similar approaches to balance autonomy and control, demonstrating the interest and relevance of the topic.

Petit and Marnewick (2019) propose a release governance approach that is adaptive with regard to their definition of a team's autonomy.

Poth et al. (2020) also also introduce the concept of team maturity and propose that higher maturity should lead to more autonomy.

Our approach aims to offer a more thorough, flexible, and seamlessly integrable solution to the challenges outlined in existing literature. Another key distinguishing feature is our focus on LSAD and security. In addition, we strive for an approach that is rigorous and systematic enough to be applied and successfully audited even in highly regulated environments, while also leveraging the benefits of agile, autonomous teams.

3 RESEARCH METHOD

Our research primarily followed a DSR approach which facilitates the creation and evaluation of artifacts that address an "unsolved problem or [...] a known problem in a more effective or efficient manner" (Hevner et al., 2004, p.81). Peffers et al. (2007) extended this concept by proposing a concrete DSR process which we use as a blueprint for our research.

We aimed to ensure rigor by building our artifacts on the results of existing literature reviews, e.g., Nägele, Schenk, and Matthes (2023), and additional literature, serving as our knowledge base.

Concurrently, we strove for relevance and practical applicability of our artifacts by employing an interview study, mainly based on Rubin and Rubin (2011) as well as Saldaña (2016), in line with the expert evaluation proposed by Peffers et al. (2012).

In the initial phase of our DSR, we engaged in unstructured interviews and informal discussions with industry partners. Additionally, we drew insights from existing literature as well as our own prior research for problem identification and motivation, as detailed in Sections 1 (Introduction) and 2 (Background and related work) of this paper.

As the second DSR step, based on the identified problem, we derived the objectives of a solution, defined in Section 4.1. We proceeded to design and develop the first iteration of our solution artifacts.

To validate our problem statement, the goals of our solution and the first iteration of our artifacts, we merged the fourth and fifth DSR steps, demonstration and evaluation, by presenting our results to expert interviewees in a first round of interviews and directly collecting feedback. This enabled us to iteratively refine our prototypes, incorporating crucial feedback and producing updated artifact versions.

Subsequently, we conducted a second interview round for the final evaluation. The final solution artifacts are presented in Section 4, with the evaluation results detailed in Section 5. Our key findings and their implications are discussed in Section 6. The communication of our findings through this paper constitutes the final phase of our DSR process.

In total, we carried out 28 interviews with 18 experts from 15 companies across five industries: finance, retail and e-commerce, consulting, entertainment, and automotive. 10 of the experts were interviewed twice and 8 experts were interviewed once, as it was not possible to interview all the experts twice due to time and capacity constraints. The average interview duration was 52 minutes, excluding initial introductions, instructions, and a final wrap-up.

We aimed for a diverse participant pool, encom-

passing ASD roles as well as security governance and audit-related positions. Specific roles that were included are Product Owner, Scrum Master, Agile Developers, Security Engineer, Security Architect, Solution Architect, Business Analyst, Security Consultant, (IT) Management Consultants, Information Security Lead, Security Officer, Security Manager, and (Security) Auditors. During selection, we prioritized experts with more extensive self-reported experience at the intersection of LSAD and security.

With the interviewees' consent, interviews were recorded and transcribed to ensure accuracy. We adhered to a cyclic approach as endorsed by Salda˜ña (2016), initially coding the data in the first cycle and filtering it more precisely in the second cycle to concentrate on the most pertinent aspects. To streamline coding, we employed the software *MAXQDA*, which facilitates efficient data processing and storage.

4 ADAPTIVE APPROACH FOR SECURITY GOVERNANCE IN LSAD

In this section, we present the resulting artifacts of our research. Those are (i) a generic organizational structure of security-related roles and (ii) a team autonomy assessment model, both integrated within (iii) an adaptive collaboration model, completed by (iv) a projection of the model onto the most relevant scaling agile frameworks. Figure 1 shows an overview of the results and the structure of this section.

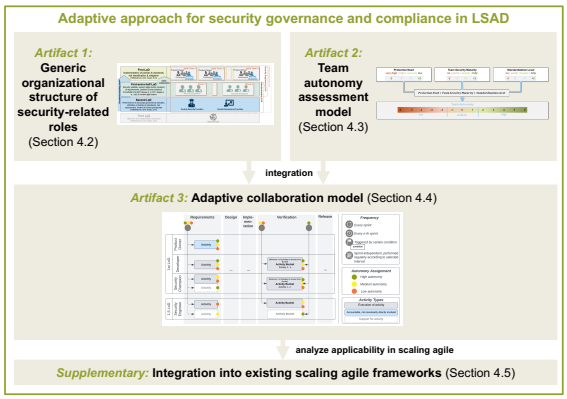

Figure 1: Overview of resulting artifacts.

Together, the resulting models form an approach for security governance in LSAD that integrates security-related roles and activities, and adapts to development team capability, as well as organizational and product requirements, thereby balancing autonomy and required control.

4.1 Goals of the Resulting Artifacts

We derived the following six objectives from the research gap and current challenges:

1. ***Achieve a clear definition of roles and security activities without being too prescriptive***, aiming to address the missing guidance on how to conduct security activities in LSAD (van der Heijden et al., 2018; Moyón et al., 2021) and tackle collaboration challenges with non-development functions (Kalenda et al., 2018).
2. ***Attain security compliance without overly rigid governance by balancing the autonomy and control tension***, avoiding controls when not required, and reducing overhead and bottlenecks.
3. ***Enable integration in scaling agile frameworks*** to increase applicability in practice, since current frameworks lack guidance on security and the control autonomy tension (Nägele et al., 2023).
4. ***Increase the security awareness and expertise of agile teams*** to address the previously reported lack thereof (Moyón et al., 2021).
5. ***Promote a responsibility shift towards agile teams*** to enable security governance procedures to be more "agile-friendly" and empower autonomous teams while ensuring compliance.
6. ***Facilitate targeted allocation of security resources*** to tackle the general shortage of security practitioners (Moyón et al., 2021).

4.2 Organizational Structure

Figure 2 illustrates the proposed organizational LSAD setup. The structure is guided by the lines of defense (LoD) model, a generic assurance approach adaptable to both security and agile contexts (Wright, 2014).

The primary alteration we introduce is the "first-and-a-half LoD", represented by security engineers (SEs). They function in a dual role, actively assisting agile teams in the first LoD while serving as the second LoD by reviewing other teams. To avoid conflicts of interest or self-examination, SEs should only assess teams and products not directly within their supportive purview. The designation as engineers emphasizes their software and security engineering expertise, enabling them to support several teams in performing security activities and increasing automation.

Following Wright's proposal, agile teams form the first LoD (Wright, 2014) and are generically composed of a product owner (PO) and developers. For simplicity, we have excluded the scrum master from our model, but their involvement is not precluded.

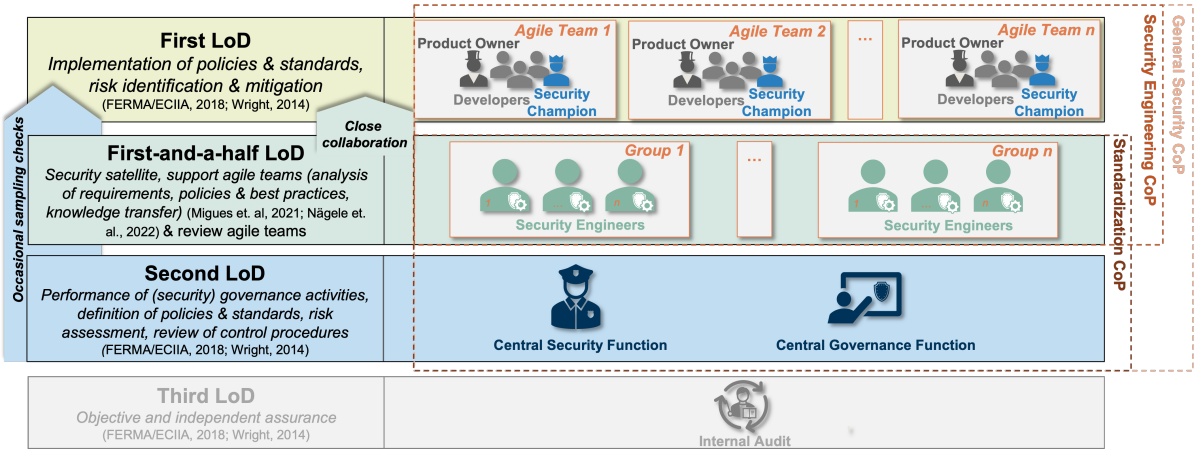

Figure 2: Generic organizational structure of security-related roles.

Moreover, we introduce the security champion (SC), a role envisioned not as a new entity but as a developer within the agile team possessing an additional commitment to security. The SC serves as the team's primary security contact point, both within the team and for external stakeholders. The roles of SEs and SCs are not entirely novel, as researchers such as Nägele et al. (2022) have already delineated common responsibilities of these roles from literature and observations in practice.

The second LoD establishes policies and standards, and confirms that the first LoD delivers acceptable work in compliance with these guidelines. In our model, the second LoD also has an enabling role, providing security tools, automated CI/CD pipelines, and training. Unlike the first and first-and-a-half LoDs, the second LoD does not directly participate in the individual teams' development activities.

The third LoD is charged with providing an independent assessment of the work performed by the first and second LoDs. Internal audits, being independent of the development process and typically conducted at random or set intervals, do not necessitate regular interaction with other LoDs in the context of our model.

4.3 Team Autonomy Assessment Model

To balance the required control and granted autonomy, we propose to systematically assess a team autonomy score based on three influencing factors: protection need, team security maturity, and standardization level.

4.3.1 Protection Need

The *protection need* is essential given the distinct risk profiles of software products. These necessitate varying types and scopes of security measures, as well as differing levels of rigor in the validation and verification processes, ultimately impacting the degree of team autonomy that can be granted.

The protection need may encompass the individual product risk and additional factors, including the organization's risk appetite and regulatory requirements. We propose that the protection need assessment is conducted by the PO, who bears product responsibility, in collaboration with an SE representing the second LoD. The SE contributes specialized knowledge to evaluate security dimensions accurately, whereas the PO ensures the consideration of the business perspective.

Our model introduces four protection need levels, omitting a middle option to preclude selection without thorough deliberation. Organizations may employ common risk analysis and predefined damage scenarios to ascertain the appropriate protection need level for a developed product. The more severe the impact of these scenarios, the higher the corresponding protection need level. Examples include non-compliance with laws or regulations, operational incapacitation, and financial harm. The configuration of damage scenarios and their impact severity per protection need level is highly organization-specific. Thus, we refrain from a more detailed definition within the model.

4.3.2 Team Security Maturity

The *team security maturity* reflects a team's capability to develop secure and regulatory-compliant software. We incorporate this factor based on the intuitive supposition that higher maturity levels warrant greater autonomy, as supported in the literature (Poth et al., 2020; Petit and Marnewick, 2019).

The comprehensive design of a maturity model to assess such a capability can be found in one of our previously published studies (Nägele et al.,

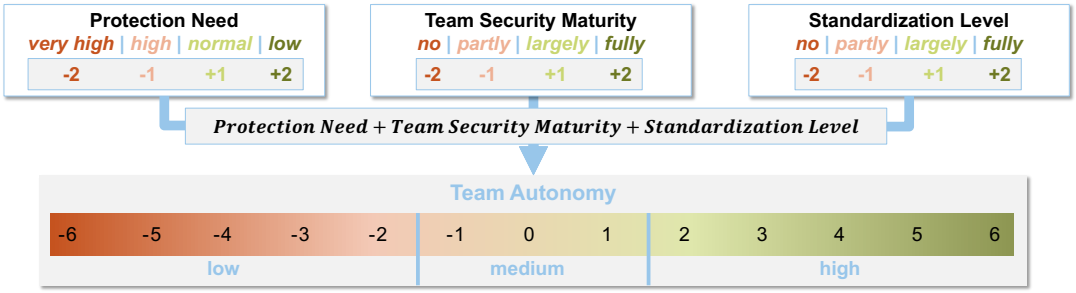

Figure 3: Team autonomy assessment calculation including exemplary thresholds.

2024). In summary, we advocate for the amalgamation of methods such as self-assessments, evaluations by external stakeholders, and the employment of (semi-)automated metrics to estimate a team's proficiency in pertinent security dimensions. Regular updates to the assessment are advised, with smaller increments to minimize overhead.

4.3.3 Standardization Level

The third determinant of team autonomy is the *standardization level*, which emerged as a prominent factor in our expert interviews.

Our proposed standardization framework comprises three pillars: (i) utilization of standardized components within the software product, (ii) the underlying infrastructure and tooling for development, deployment, and delivery, and (iii) security-related activities.

Reusable components may address cross-team security requirements, such as authentication, authorization, or secret management. Standardized infrastructure facilitates pre-tested and configured environments for teams, exemplified by customizable pipelines for building, testing, and deploying software iterations on trusted platforms. Employing common standards for these pipelines, such as the same static and dynamic security testing tools with organization-optimized rulesets, enhances comparability, quality, and knowledge sharing.

Standardizing security activities through clear guidance on practices like security code reviews or threat modeling enables greater autonomy. The SE's role is crucial in assessing and guiding teams.

Although standardization may be evaluated as part of the team maturity assessment, its significance warrants separate consideration.

4.3.4 Team Autonomy Score

To integrate all three factors into a team autonomy score, our interview findings suggest a preference for a straightforward calculation method to enhance practical applicability. Each factor is assigned one of four proficiency levels determined through prior assessments, with absolute values representing *very negative (-2)*, *negative (-1)*, *positive (+1)*, or *very positive (+2)* influences on team autonomy. The team autonomy score is obtained by summing these values, with thresholds demarcating low (< -1), medium (> -1 *and* ≤ 1), and high (> 1) autonomy.

Figure 3 depicts the logic, scale, and thresholds. This scale enables compensation for unfavorable conditions; for instance, a high protection need, typically necessitating higher control, can be counterbalanced by high team security maturity and extensive standardized component usage, resulting in high autonomy.

This compensation logic aligns with our overarching goal of fostering high team autonomy, removing bottlenecks, and streamlining processes long-term while motivating teams to improve. Without compensation, a high protection need level would automatically yield low autonomy, reducing incentives for security training and maturity improvement. Organizations can tailor the calculation logic to their requirements and context.

4.4 Adaptive Collaboration Model

Our adaptive collaboration model integrates development-related security activities and assigns them to roles defined in the generic organizational setup. It is adaptive by adjusting role assignments and task frequencies based on team autonomy.

Figure 4 depicts the generic structure of the model, featuring key components such as security activities (grouped by phases), security-related roles, and the frequency of activity execution. The model illustrates the impact of team autonomy on the assignment of roles and the frequency of security activities, represented through color-coded markers. The requirement for a team to undertake a marked activity is contingent on its respective level of autonomy. Unmarked activities for a specific level of autonomy

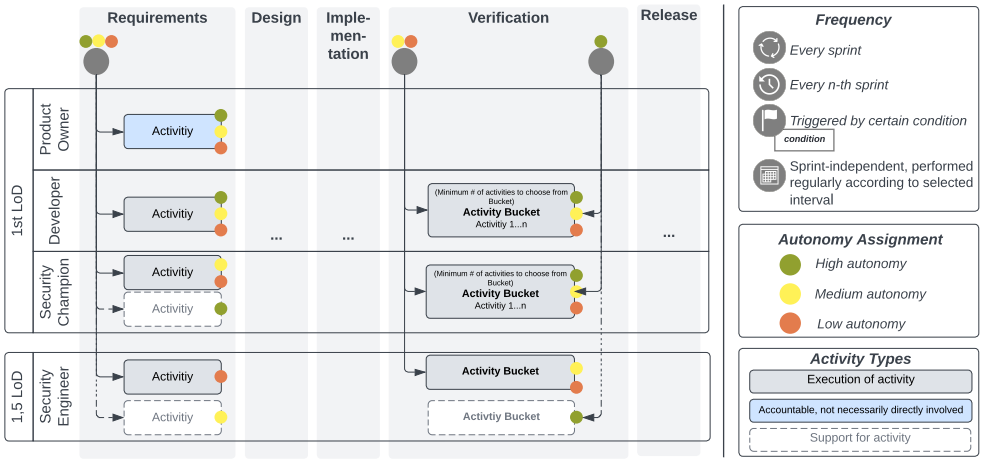

Figure 4: Generic structure of the collaboration model.

are optional and can be disregarded, provided that the team has an adequate autonomy level. However, these assignments should be customized to the specific organization. Generally, we propose that higher autonomy correlates with reduced collaboration with SEs, lower reliance on the SC, and a minimized mandatory frequency of security activities.

To structure and categorize the activities, we derived a common denominator from secure development life cycles in the literature (Boldt et al., 2017; Microsoft, 2012). The model differentiates between *active execution*, *accountability without imposed direct involvement*, and *supporting an activity* (see Figure 4).

It further distinguishes between concrete activities and *activity buckets*, a concept adapted from Microsoft (2012) that enables grouping similar activities. Buckets allow teams to systematically distribute necessary activities over time without having to perform them for every deployment or sprint. Organizations can determine the number of bucket activities to be performed per sprint and establish a cycle to ensure all practices within a bucket are eventually executed.

Concrete examples of security activities are threat modeling, penetration testing, security code and architecture reviews or audits, and implementing improvements according to the findings of static (SAST) and dynamic (DAST) application security testing tools. The selection of specific activities is outside the scope of the model. However, one of our previously published studies provides more in-depth guidance on selecting appropriate security activities in LSAD environments (Nägele et al., 2023).

The model also accommodates custom execution frequencies for individual activities. While some activities might be required every sprint or every n-th sprint, others are only required if a certain trigger is reached, such as a deployment to production. Additionally, certain activities might not be tethered to sprint timelines but rather to designated timespans, e.g., dictated by regulatory requirements.

While the security activities are sequentially categorized, the model should not be interpreted as a linear process. Instead, it serves as an activity catalog, guiding role involvement in necessary security activities. It is essential that team members identify the current category of work to determine the required security measures for that development stage.

4.5 Integration with Scaling Agile Frameworks

This section summarizes the compatibility of our approach with six of the most used scaling agile frameworks in practice (Uludağ et al., 2022): The Scaled Agile Framework (SAFe), Scrum@Scale, Large-Scale Scrum (LeSS), Spotify model, Disciplined Agile (DA) and Nexus.

Overall, our analysis did not discover any irreconcilable conflicts that might prevent the adoption of our model, though certain frameworks necessitate specific considerations.

One potential conflict that might require a softening of framework guidelines is incorporating specialist roles in agile teams. Whereas DA and SAFe are more in line with our recommendation to introduce SCs by describing a specialized role in every team (Ambler and Lines, 2020) or at least the opportunity to include specialists (Knaster and Leffingwell, 2020), other frameworks do not discuss such roles.

In our perspective, these additional roles offer significant value, provided organizations avoid reverting to outdated patterns of specialized roles that do

not contribute to development. In our model, the SC is an integral team member, contributing equally to value creation but with an additional 'hat' for security. However, the SC role may be a transitional mechanism until the team achieves sufficient security maturity, at which point it may be redundant. LeSS opposes the introduction of additional roles, advocating instead for 'travelers' – specialists who temporarily join a team to facilitate knowledge transfer before rotating to another (Larman and Vodde, 2016). Organizations using LeSS could deploy SCs as travelers to maintain compatibility.

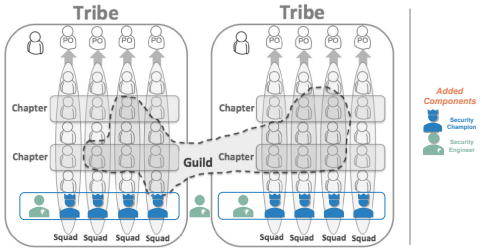

Figure 5: Spotify model exemplary integration.

Introducing team-external roles supporting agile teams is more common and prevalent in all examined frameworks, except for the Spotify model. However, with its concept of chapters and guilds, the Spotify model aligns well with our model, as depicted in Figure 5.

In Nexus, the SE can be included in the integration team, as shown in Figure 6.

The most considerable ambiguity arises concerning the proposed integration of the second LoD. Scrum@Scale includes similar central functions, such as legal and compliance. Other frameworks do not explicitly feature such centralized teams, necessitating a more tailored transfer. Nonetheless, in our model, the second LoD serves as a valuable auxiliary function rather than a core component and could be substituted by other mechanisms. Consequently, we do not perceive this as a significant impediment to the integration of our model.

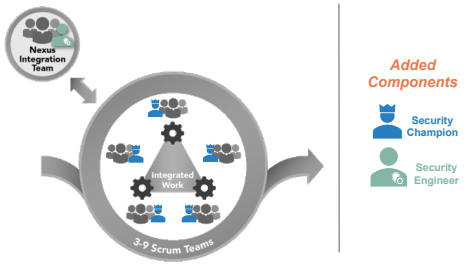

Figure 6: Nexus exemplary integration.

5 EVALUATION

This section summarizes the results of the expert evaluation of our research artifacts. In general, the results are in favor of our approach. Nevertheless, the evaluation also brought interesting controversial aspects to light.

5.1 Organizational Setup Evaluation

In general, all experts considered the proposed organizational setup, including the LoDs, to be clear, rational, and integrable into established systems in their organizations. However, one expert raised concerns about the suitability for smaller organizations due to the additional roles.

The concept of SCs was generally favored, with one exception. Experts suggested role assignment should be affinity-based, not knowledge-based, and proposed SCs act as a central contact for agile developers as well as the second LoD and auditors. Consequently, we expanded the SC's role to include this suggestion. A disputed point was the necessity of an SC or SE in every team. Some experts advocated for a mechanism to assess this necessity, whereas others emphasized only the need for variable allocated capacities. Our model presumes baseline security requirements that necessitate an SC in every team, which most experts agreed with. However, we recognize the mandatory SCs as a limitation that requires further research on its impact on applicability.

The experts stated that successful SC implementation necessitates clearly defined incentives, training roadmaps, and long-term goals. Adequate capacity for their tasks and training is crucial.

Experts agreed on employing SEs to support agile teams and act as reviewers for teams that do not already receive their support. The introduction of the first-and-a-half LoD and its associated SEs received particular praise, especially for ensuring duty separation and in-depth reviews. However, offloading team responsibility to SEs was identified as a potential risk, which our model mitigates by defining SEs as supporters, not primary actors.

The experts confirmed including central governance and security functions in the second LoD to facilitate cross-team collaboration and balance business needs and security requirements. They also supported the translation of generic policies into concrete security standards by central security teams, with SEs assisting in their practical implementation. However, they also discussed alternatives to central teams, such as a security architect role or self-organization through communities, which presuppose a high level

of organizational maturity.

Experts agreed with not explicitly including the third LoD, as active involvement would conflict with its audit function. However, they noted interest in the model's promotion of continuous compliance and potential auditing culture shift.

Communities of Practice (CoPs) were generally endorsed for their role in knowledge sharing, although incentives for participation were seen as critical for their success. One expert explained that "enabling knowledge sharing is one of the most important aspects in agile [environments]", thereby stressing the importance of security-focused CoPs.

While deeming our model applicable, one expert suggested a more generic version to account for other types of non-functional requirements.

5.2 Team Autonomy Assessment Evaluation

The experts considered the assessment of team autonomy and influencing factors feasible and valuable, provided that adequate resources such as SEs are available. One expert noted that the assessment aligns well with agile principles and methods, e.g., retrospectives. Another expert voiced concerns about allowing low-autonomy teams to handle high-protection-need projects, suggesting to remove the protection need from the autonomy assessment and assigning the respective projects according to the team autonomy instead. We decided against such a removal but acknowledge the possibility of adding individual constraints or actions to the model, e.g., changing the team-product assignment in case of a "low" assessment result.

An auditor proposed that a maturity score could prove beneficial for tracking the impact and progression of security initiatives over time. However, the expert observed that such scores typically do not suffice to derive suitable measures for enhancing security maturity, potentially leading to overlooked deficiencies, especially when a team's overall maturity rating is high yet has a significant shortfall in a particular area.

No additional factors affecting team autonomy were identified during the evaluation, but a consensus was reached that organizational risk appetite influences protection need and should not be assessed separately. Organizations adopting such an approach should be able to clearly communicate in an audit why certain teams are assumed to have a specific level of security maturity and how the resulting governance approach aligns with the general risk management.

The inclusion of team security maturity was universally valued, particularly when evaluated using objective measurements. While self-assessments were deemed valuable for lean, self-dependent governance, their subjectivity was viewed as a potential drawback.

The standardization level was generally deemed important, despite potential loopholes and workarounds in highly standardized systems. Experts suggested additional aspects, such as the proportion of outsourced components, since these might introduce risks that development teams do not directly control.

5.3 Adaptive Collaboration Model Evaluation

The experts agreed that increased autonomy serves as a compelling incentive for teams to enhance their security focus, thereby facilitating the transition of security responsibilities to the teams. They regarded the model as flexible and adaptable, providing sufficient guidance while preserving room for customization.

One expert emphasized the positive impact of collaborative security activities such as threat modeling and stated that "[...] once you show them how something can go wrong, they are going to be self-motivated to make the thing [the threat] go away. They are going to educate themselves."

Another expert stated that the model's adaptability makes it a "valuable resource-saving approach" by focusing efforts on teams with less autonomy addressing the important lack of security resources issue.

Nonetheless, auditors emphasized the importance of documenting the completion of security activities for compliance, which should be ensured across all teams, irrespective of maturity. Auditors also noted that while the adaptive approach does not conflict with common security standards, initial audits may necessitate closer examination due to its novelty.

The categorization of activities is deemed useful, adding to the model's flexibility. One expert highlighted the inclusion of triggered activities, as it is something the expert has "not seen so far in any other model", but thinks that they provide great value.

We offered illustrative configurations to enhance the model's comprehensibility, encompassing specific security activities. However, the evaluation highlights that the selection and configuration of activities largely depend on the specifics of the organizations.

Overall, the experts were satisfied with the level of complexity, for example stating that "the model is not over-engineered" and that "[...] being too detailed would be the pitfall of the model."

However, while appreciating the model's ability to distill a complex issue, experts expressed concerns

about managing the remaining complexity - in particular, conveying the model's configuration to the respective teams considering their team autonomy. We showcased potential solutions for this challenge by using the web-based diagramming tool "Lucidchart" for demonstration. Its simple interactive elements allow to create filters for different maturity levels and a process diagram that adapts to the selected filters, reducing the visualization complexity and enhancing comprehensibility - a factor critical for the model's practical applicability.

5.4 Evaluation of Framework Integration

Expert opinions on incorporating the artifacts into existing scaling agile frameworks varied. Some experts saw merit in offering integration guides, arguing that such guidance could streamline adaptation for companies utilizing these frameworks. Others recognized the potential utility but considered the integration straightforward and thus would not prioritize it. One expert highlighted the possible resistance of agile practitioners to the introduction of specialized roles, stressing the importance of illustrating the congruence between our role model and these frameworks.

6 DISCUSSION

6.1 Key Findings

To answer our RQ and address the identified research gap, we created artifacts that guide in balancing the control and autonomy tension in LSAD. In the following, we present four key findings of our research.

First, the expert evaluation finds that our adaptive approach, anchored in team autonomy assessments, forms a valuable basis for resource-efficient governance for security compliance within LSAD. According to our interviews, the current modus operandi in large-scale enterprises leans towards top-down governance procedures. However, we advocate for a transformation towards autonomy and self-governance to facilitate sustainable security integration into LSAD. Organizations can leverage team autonomy to adapt their governance process and avoid having controls in place when circumstances do not call for them, aiming to ensure security compliance while reducing governance bottlenecks. This approach improves the compatibility of security governance and compliance efforts with agile methods at scale, a key challenge reported in existing research (Nägele et al., 2023).

Second, the experts assessed the collaboration model as valuable. The model provides a customizable distribution of security activities among roles and iterative development phases, thereby offering clear guidance on the collaboration between agile teams and security experts. By incorporating visualization and a filtering functionality, the model enhances its usability and boosts transparency. Team autonomy, which serves as an input to the model, helps to determine the roles that participate in an activity and the frequency of specific activities, thereby achieving a desirable degree of adaptivity. We recommend a reduced collaboration with team-external stakeholders and security specialists for teams with high autonomy, and fewer prescribed security activities and audits. By shifting responsibility to teams, the model aims to conserve scarce security resources while encouraging teams to prioritize security and achieve faster delivery. We thereby tackle challenges described in previous literature, such as the shortage of security practitioners (Moyón et al., 2021) and the need for guidance on integrating security activities in LSAD (Edison et al., 2021; Dännart et al., 2019).

Thirdly, during expert interviews, some intriguing perspectives on the rigidity of our approach emerged. Initially, a few auditors preferred more structure and control, while some agile practitioners found it too rigid and process-based. This divergence in perceptions affirmed the relevance of the tension we aim to reconcile. Through further dialogue, iterations, and the incorporation of the expert feedback into our models, both groups started to appreciate the advantages of our approach. Governance and auditing roles particularly valued the systematic and traceable method for granting autonomy. At the same time, agile practitioners saw value in the approach because it empowers experts to hone and refine their skills, rewarding expertise with greater autonomy and accountability.

Lastly, we acknowledge that our approach is extensive, and its implementation within an organization may require significant effort. However, our approach could also inspire incremental changes to existing established procedures within organizations. Our expert interviews also corroborated this, sparking considerable interest in integrating the perspective of team autonomy within their organizations.

6.2 Limitations

To yield robust results and artifacts, we incorporated expert interviews into the development and final evaluation of our work. Recognizing the potential impact of expert representativeness on the validity of our findings (Miles et al., 2014), we prioritized experts

with extensive experience and diverse backgrounds.

Additionally, we acknowledge the potential influence of researchers on the objectivity of the collected data and its analysis (Miles et al., 2014). To mitigate this, we followed the interview questionnaire as closely as possible during the semi-structured interviews and employed cyclic coding in our analysis. Transcription of the interviews further limited the influence of researcher-inherent bias.

Another notable limitation of this study lies in the absence of experiments and practical applications of our approach due to the complexity of the research topic. However, such investigations are currently in progress, even extending to cross-cutting concerns beyond security, such as software architecture and user experience.

7 CONCLUSION

The increasing adoption of LSAD, coupled with the growing significance of security, necessitates approaches to achieve a balance between team autonomy and organizational control. Existing literature recognizes this tension and its resultant challenges, but comprehensive solutions remain scarce. To address this gap, we employed a DSR approach to develop solution artifacts. We combined this approach with an expert interview study to improve our artifacts and ensure practical relevance and applicability. Our approach aspires to harmonize governance with LSAD by using a team autonomy assessment as a determinant of role responsibilities and security activity frequency. Our central recommendation is that more capable and mature teams should receive more autonomy, based on a documented and transparent assessment and process model. This solution balances the granted autonomy and required control while preserving compliance and auditability. The expert evaluations generally endorse our approach while illuminating areas for enhancement and additional research opportunities. Future experiments should scrutinize the practicality of our approach, perhaps focusing initially on a select few teams to increase feasibility before scaling to larger environments. Prospective adaptations of the approach could also encompass other cross-cutting concerns and non-functional requirements beyond security.

ACKNOWLEDGEMENTS

This work has been supported by the German Federal Ministry of Education and Research (BMBF) Software Campus grant 01IS17049.

REFERENCES

Ambler, S. W. and Lines, M. (2020). *Choose Your WoW: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working*. PM Institute.

Bartsch, S. (2011). Practitioners' perspectives on security in agile development. In *Sixth International Conference on Availability, Reliability and Security (ARES)*, pages 479–484, Piscataway, NJ. IEEE.

Bell, L., Bird, J., Brunton-Spall, M., and Smith, R. (2017). *Agile application security: Enabling security in a continuous delivery pipeline*. O'Reilly Media, Sebastopol, CA.

Boldt, M., Jacobsson, A., Baca, D., and Carlsson, B. (2017). Introducing a novel security-enhanced agile software development process. *International Journal of Secure Software Engineering*, 8(2):26–52.

Dännart, S., Moyón, F., and Beckers, K. (2019). An assessment model for continuous security compliance in large scale agile environments. In *Advanced Information Systems Engineering*, pages 529–544, Cham, Switzerland. Springer.

Dikert, K., Paasivaara, M., and Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108.

Edison, H., Wang, X., and Conboy, K. (2021). Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering*, pages 1–23.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in IS research. *MIS Quarterly*, 28(1), 75–105.

Julisch, K. (2008). Security compliance: The next frontier in security research. *Proc. of the new security paradigms workshop*, 71–74.

Kalenda, M., Hyna, P., and Rossi, B. (2018). Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10):p. 1954.

Knaster, R. and Leffingwell, D. (2020). *SAFe 5.0 Distilled: Achieving Business Agility with the Scaled Agile Framework*. Addison-Wesley, Boston.

Larman, C. and Vodde, B. (2016). *Large-scale Scrum: More with LeSS*. Addison-Wesley, Boston and Amsterdam and London.

Microsoft (2012). Security development lifecycle - sdl process guidance version 5.2.

Miles, M. B., Huberman, A. M., and Saldaña, J. (2014). *Qualitative data analysis: A methods sourcebook*. SAGE, Thousand Oaks Californinia, 3 edition.

Moyon, F., Almeida, P., Riofrio, D., Mendez, D., and Kalinowski, M. (2020). Security compliance in agile software development: A systematic mapping study. In *46th Euromicro Conference on Software Engineering*

and Advanced Applications (SEAA), pages 413–420. IEEE.

Moyón, F., Méndez, D., Beckers, K., and Klepper, S. (2021). How to integrate security compliance requirements with agile software engineering at scale? In *Product-Focused Software Process Improvement*, pages 69–87, Cham, Switzerland. Springer.

Nägele, S., Korn, L., and Matthes, F. (2023). Adoption of information security practices in large-scale agile software development: A case study in the finance industry. In *Proc. of the 18th Int. Conf. on Availability, Reliability and Security*, ARES '23, New York, NY, USA. Association for Computing Machinery.

Nägele, S., Schenk, N., and Matthes, F. (2023). The current state of security governance and compliance in large-scale agile development: A systematic literature review and interview study. In *IEEE 25th Conference on Business Informatics (CBI)*.

Nägele, S., Watzelt, J.-P., and Matthes, F. (2022). Investigating the current state of security in large-scale agile development. In *23rd Int. Conf. on Agile Software Development (XP)*, pages 203–219. Springer.

Nägele, S., Watzelt, J.-P., and Matthes, F. (2024). Assessing team security maturity in large-scale agile development. In *Proc. of the 57th Hawaii Int. Conf. on System Sciences*, pages 7259–7269.

Newton, N., Anslow, C., and Drechsler, A. (2019). Information security in agile software development projects: A critical success factor perspective. In *27th European Conference on Information Systems (ECIS)*, pages 1 – 17.

Peffers, K., Rothenberger, M., Tuunanen, T., and Vaezi, R. (2012). Design science research evaluation. In *Design Science Research in Information Systems. Advances in Theory and Practice*, volume 7286, pages 398–410. Berlin, Heidelberg.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77.

Petit, Y. and Marnewick, C. (2019). Earn your wings: A novel approach to deployment governance. In *Agile Processes in Software Engineering and XP – Workshops*, volume 364, pages 64–71, Cham, Switzerland. Springer.

Poth, A., Jacobsen, J., and Riel, A. (2020). Systematic agile development in regulated environments. In *Systems, Software and Services Process Improvement*, volume 1251, pages 191–202, Cham. Springer.

Rindell, K., Hyrynsalmi, S., and Leppänen, V. (2018). Aligning security objectives with agile software development. In *19th Int. Conference on Agile Software Development*, pages 1–9, New York, NY, USA. ACM.

Rubin, H. J. and Rubin, I. S. (2011). *Qualitative Interviewing: The Art of Hearing Data*. SAGE, 3 edition.

Saldaña, J. (2016). *The coding manual for qualitative researchers*. SAGE, Los Angeles and London, 3 edition.

Stray, V., Moe, N. B., and Hoda, R. (2018). Autonomous agile teams: Challenges and future directions for research. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pages 1–5, New York, NY, USA. ACM.

Tayaksi, C., Ada, E., Kazancoglu, Y., and Sagnak, M. (2022). The financial impacts of information systems security breaches on publicly traded companies: reactions of different sectors. *Journal of Enterprise Information Management*, 35(2):650–668.

Uludağ, Ö., Philipp, P., Putta, A., Paasivaara, M., Lassenius, C., and Matthes, F. (2022). Revealing the state of the art of large-scale agile development research: A systematic mapping study. *Journal of Systems and Software*, 194.

van der Heijden, A., Broasca, C., and Serebrenik, A. (2018). An empirical perspective on security challenges in large-scale agile software development. In *12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 1–4, New York, NY. ACM.

Williams, P. (2001). Information security governance. *Information Security Technical Report*, 6(3), 60–70.

Wright, C. (2014). *Agile Governance and Audit*. IT Governance Publishing, Cambridgeshire.

Proceedings of the 57th Hawaii International Conference on System Sciences | 2024

Assessing Team Security Maturity in Large-Scale Agile Development

Sascha Nägele
Technical University of Munich
sascha.naegele@tum.de

Jan-Philipp Watzelt
Technical University of Munich
jan-philipp.watzelt@tum.de

Florian Matthes
Technical University of Munich
matthes@tum.de

Abstract

Organizations struggle to balance agile team autonomy and strict security governance in large-scale agile development environments. In particular, conventional top-down IT governance mechanisms often conflict with the desired autonomy of decentralized agile teams. Our research presents a novel approach to resolve the tension between security governance and development agility: a criteria-based security maturity assessment that enables greater autonomy for mature agile teams. Leveraging design science research, a literature review, and an interview study, we introduce two key contributions: a criteria catalog for evaluating a team's capabilities and a team security maturity model. Our expert evaluation confirms their value for systematically assessing the teams' capabilities to deliver secure and compliant applications, allowing organizations to grant more autonomy to mature teams and prioritize supporting lower-maturity teams. Future work could go beyond expert interviews and implement and evaluate the team security maturity model through a case study or experiments.

Keywords: Large-scale agile development, team maturity, security, governance, compliance

1. Introduction

As organizations increasingly adopt scaled agile methods to achieve benefits such as faster time-to-market (Uludağ et al., 2021), they simultaneously face mounting security challenges due to escalating threats and more stringent legislation (Rindell et al., 2021). This dynamic creates a unique set of challenges within large-scale agile development (LSAD) environments (van der Heijden et al., 2018), fueling a tension between the necessity for centralized security governance and the desire for autonomy among agile development teams (Horlach et al., 2018; Nägele et al., 2022).

A potential solution to alleviate this tension is to evaluate the team maturity and capability, and grant autonomy accordingly (Poth et al., 2021). More mature development teams may be more capable of self-governing their security posture, requiring less top-down control (Nägele et al., 2022).

However, concrete guidance on how to assess the security maturity of agile development teams is scarce. None of the maturity models identified in the literature focus on the security capability of agile teams, which is why we also considered models from non-academic sources. These existing maturity models either focus on organization-wide assessments and therefore are not usable for maturity assessments of individual teams, or capture team security maturity in the form of activity checklists rather than measuring the maturity of each activity. Furthermore, they rely solely on single assessment types like self-assessments and do not specifically address the autonomy-control friction in LSAD.

To fill this gap, our research question (RQ) is:

RQ: How can a maturity model be designed and implemented to assess a team's capability to develop secure and security-compliant applications in LSAD?

We employ a design science research (DSR) methodology to create a team security maturity model that addresses the identified gap in academic literature and the shortcomings of existing models. To increase the rigor and relevance of our artifact, we conducted a systematic literature review (SLR) and an interview study within our DSR approach.

Our results provide practical guidance on how to assess the security capability of agile teams in LSAD

URI: https://hdl.handle.net/10125/107257
978-0-9981331-7-1

settings and balance the autonomy control tension.

The paper is structured as follows. Sections 2 and 3 cover background and related work. Section 4 explains our research method. Section 5 introduces our ten criteria for team security maturity that build the basis for our maturity model, presented in Section 6. Section 7 summarizes the results of our evaluation, and Sections 8 and 9 discuss and conclude our findings.

2. Theoretical background

According to Dikert et al. (2016), *LSAD* environments comprise at least 50 people or six teams. We selected this definition for our study due to its systematic derivation from a mapping study and its use among other LSAD researchers (Uludağ et al., 2021).

In our study, *security* denotes a subset of *information security*, which aims to "ensure business continuity and minimize business damage by preventing and minimizing the impact of security incidents" (von Solms, 1998, p. 224). This involves safeguarding the availability, integrity, and confidentiality of information and systems (Bell et al., 2017). Given our research emphasis on development teams, our focus within information security is on secure software development and application security, targeting risk minimization through technical and organizational measures during software application development and operation (Bell et al., 2017). *Security compliance* refers to adherence to security requirements (Julisch, 2008), which may stem from two sources: external, such as regulatory bodies or industry standards, and internal, from policies and guidelines.

To assess the capability of teams to fulfill such requirements, we propose using a maturity model. The most frequently cited type of maturity model in literature is the Capability Maturity Model Integration, a derivative of the Capability Maturity Model (CMM) (Wendler, 2012). However, so-called *maturity grids* represent an important alternative to these models (Maier et al., 2012). They can be used both as an assessment and as an improvement tool and differ from CMM-based models in the aspects of work orientation, mode of assessment, and intention (Maier et al., 2012).

3. Related work

We identified six existing maturity models relevant to our objective of assessing team security maturity. We excluded models aiming to assess the agility of development teams from our research scope, since they do not provide actionable guidance for our research goal to assess security capabilities. Only two of the six models originate from academic literature, but they both do not focus on security. Due to this scarcity of applicable models, we included non-academic sources as explained in Section 4, which resulted in four additional relevant models. We categorize them into two types: organization-level and team-level models.

Organization-level security maturity models, namely the Software Assurance Maturity Model (SAMM) (OWASP Foundation, 2022) and the Building Security in Maturity Model (BSIMM) (Synopsys, 2021) are primarily oriented towards whole-organization assessment, deviating from our study's focus on team assessment. However, we deem these models relevant to our study because they offer guidance transferable to the context of individual teams and provide instructions for integrating maturity models into overarching organizational processes.

Poth et al. (2021) introduce a team maturity model designed to assess team performance, adhering to the principle that higher maturity leads to more autonomy. While not primarily focused on security, the model incorporates security as one of its pillars within its fundamental structure of pillars, domains, and topics.

Pagel's (2020) DevSecOps Maturity Model (DSOMM) provides security measures and prioritization when deploying DevOps teams. The model utilizes dimensions, sub-dimensions, and maturity levels. Dimensions represent categories, such as "build and deploy", and sub-dimensions further specify those dimensions, e.g., "patch management". DSOMM describes sets of activities required to achieve a certain maturity, whereas our model proposes examining maturity for each capability.

The Security Belts model (AppSecure.nrw, 2021), inspired in part by DSOMM and SAMM, similarly structures security capabilities and assesses team maturity. As in our model, its primary focus is on assessing the maturity level of teams for secure software development. The structural approach of this model assigns maturity levels based on performed activities, akin to a checklist. In contrast, our proposition involves investigating maturity levels within each topic.

Finally, Britto et al. (2016) describe in a case study how Ericsson uses team maturity levels to adjust the responsibility of distributed development teams, among other findings. Although not directly linked to security, this study exemplifies practical applications of team maturity in large-scale environments.

To summarize, our proposed model distinguishes itself by employing a maturity grid system with textual descriptions for each topic's maturity level. Moreover, our model incorporates a multi-source approach, combining self-assessments, external assessments, and

automated metrics to bolster result validity. Uniquely, our model's creation and evaluation occur specifically within the context of security and LSAD, a feature not found in the other models. Finally, we close a gap in academic literature, since all the presented models focused on the security maturity of development teams stem from non-academic sources.

4. Research method

To address our RQ, we utilized the DSR process of Peffers et al. (2007) because it enabled us to systematically create and evaluate a solution artifact, the TSMM, to address an identified problem, in our case, the autonomy control tension in security and LSAD. Rigor was ensured by conducting an SLR while relevance was obtained through interviews and workshops with industry experts. We chose these methods because they are typical examples of suitable methods to create maturity models in the context of IT (Becker et al., 2009).

Further details of these methods are provided below, and additional information can be found in our research protocol and supplementary material (Nägele et al., 2023).

4.1. Systematic literature review

Our SLR aimed to identify influencing factors for developing secure and security-compliant applications in LSAD, as well as existing team security maturity models. We divided our SLR into four phases. These phases are derived from Webster and Watson's (2002) suggestions for structuring a literature review and described in the following.

Phase 1 - Foundation: We chose our databases - ACM, Web of Science, Science Direct, IEEE, Google Scholar, and Scopus - based on initial results and the selection of other LSAD researchers (Dikert et al., 2016). Subsequently, we formulated and iteratively refined our search string, resulting in: *('large-scale agile' OR 'agile at scale') AND 'software' AND ('team' OR 'teamwork') AND ('maturity' OR 'assessment' OR 'self-assessment' OR 'capability' OR 'quality' OR 'security')*.

Phase 2 - Synthesis and analysis: In the second phase, we applied our search string, aggregated the results, and initiated the analysis process by eliminating duplicates and screening titles for relevance. For Google Scholar, we used the additional "exclude citations" and "review article" filters to reduce the number of results. From the resultant 138 publications, we evaluated abstracts and applied exclusion criteria prior to full-text review. We excluded publications predating the 2001 Agile Manifesto, non-English full texts, and publications outside of journals or conferences. This resulted in 51 papers, of which we deemed ten relevant based on full-text analysis.

Phase 3 - Extension: We enriched our literature set via backward and forward searches. Backward snowballing, applied to the background and related work sections of the most relevant publications, identified 28 new candidate articles. Forward snowballing involved using Google Scholar to track all articles citing the four publications we deemed most relevant, yielding 44 more candidate publications. Following the procedure of the second phase, we merged and screened the new titles from both methods. Since we could not find any maturity models that assess the security competence of development teams, and only two maturity models that are at least partially relevant to our research objective, we decided to include non-academic sources. This led to four additional models for our related work discussed in Section 3. We consider this reasonable in light of the application-oriented nature of our research. Overall, the third phase led to the inclusion of 12 additional publications.

Phase 4 - Evaluation: We identified and selected recurring best practices on how to evaluate the security capability of development teams from our 22 identified publications and documented and categorized the results in a concept matrix, as proposed by Webster and Watson (2002). The concept matrix is the basis for our written analysis and presentation of our results in Section 5.

4.2. Interviews and workshops

We conducted expert interviews to demonstrate and evaluate our artifacts to ensure practical relevance and applicability. We used the ACM standard (ACM SIGSOFT, 2023) for qualitative surveys to guide our interview study. In the following, we will briefly explain the study design and data collection and analysis.

Study design: We created and used a semi-structured questionnaire because it provides stringent interview guidance while allowing enough freedom in the answers and the possibility for individual adjustments during the interview, e.g., based on the experts' expertise (Döring et al., 2016). The experts were acquired through LinkedIn and our research network. The interviewee roles included (information) security consultants and architects, secure development experts, a senior secure development researcher, and a software quality and governance expert. Represented industries are automotive, finance, engineering, and media.

Data collection: We conducted synchronous interviews with one participant at a time using online videoconferencing tools. In total, we conducted 12 interviews. The average interview duration was 50

	Criteria	Example assessment questions
Non-Associated SDLC	Awareness	Does the team understand the relevance of security (compliance) for their product?
	Composition	Does the team include at least one security expert, such as a Security Champion, for support?
	Knowledge	Is the team familiar with the security best practices and policies applicable to their product? Are there established mechanisms for knowledge sharing within the team?
	Training	Does the team (regularly) engage in improvements on security-related topics?
	Collaboration	Does the team frequently engage with security experts outside the team?
Associated SDLC	Activities	Does the team have procedures to identify security threats and vulnerabilities, both manually (e.g., code reviews) and through tools (e.g., SAST and DAST)?
	Development	Does the team promptly address identified vulnerabilities? Does it establish and uphold quality gates?
	Documentation	Does the team create consistently structured security documentation with minimal overhead?
	Product	What is the level of security quality and compliance of the products developed by the team?
	Responsibility	Does the team consider security as a requirement and assign corresponding responsibilities?

Figure 1. Overview of the team security maturity criteria

minutes. The average security experience of our interviewees was six years, and seven years in scaling agile. With the consent of the participants, we recorded the sessions and transcribed them.

Data analysis: We used the approach by Kuckartz (2016) to analyze our interview study data because it offers a deductive-inductive classification of interview statements. We conducted the content structuring analysis of our interview transcripts using the qualitative data analysis software *MAXQDA*.

As a supplementary method, we organized two workshops with two secure development experts active in LSAD to facilitate in-depth discussions of our solution artifacts. These workshops, each spanning three hours, offered the possibility for a more comprehensive discourse compared to the interviews and coined the selection of the security maturity criteria and related recommendations, as well as the structure and content of the TSMM.

5. Team security maturity criteria

We propose ten criteria for assessing team security maturity, laying the groundwork for our maturity model. Inspired by Bishop and Rowland's (2019) literature review structure on agility and security, we categorize the criteria into two groups: non-associated and associated with the software development lifecycle (SDLC) phases, each containing five criteria. Both categories are crucial for a mature team, which we characterize not only by security competencies during development, but also by overarching criteria like security awareness, team composition, and collaboration with other roles, which is especially important in the context of LSAD.

Figure 1 provides an overview of the criteria and exemplary measurement questions. In the following, we explain the criteria by presenting the relevant theory and our recommendations and suggestions, influenced by our expert interviews.

5.1. Criteria non-associated with the SDLC

Awareness: Teams often view security as nonessential in software development (van der Heijden et al., 2018). However, understanding its importance fosters responsible handling of security requirements and boosts motivation (Bodin & Golberg, 2021). Although interconnected, we consider awareness and security knowledge separate criteria to emphasize their importance. The experiences of our interviewees reveal crucial differences in the prioritization of security when teams comprehend specific risks linked to their product rather than merely fulfilling obligations. This understanding enables careful selection of suitable security measures, potentially reducing long-term efforts. To assess maturity, we suggest inquiring whether teams understand the product-specific security relevance of their system.

Composition: A team's composition, such as the inclusion of a security champion (SC), significantly influences its capacity to develop secure applications (Jaatun & Soares Cruzes, 2021). The SC, typically a software engineer specialized in security and often instrumental in raising awareness and quality, guides the team with security-related tasks like risk analysis or security code reviews. Continuous training and

experience sharing are crucial for the SC (Pagel, 2020), who often collaborates with other teams in security communities (Nägele et al., 2022).

We suggest evaluating teams based on the presence of at least one security-knowledgeable developer and their commitment to continuous learning and ability to share knowledge with the team.

Knowledge: Developing high-quality products necessitates knowledge of relevant best practices, standards, and their application (Poth, Kottke, & Riel, 2021). Mature teams incorporate security early by understanding design principles and adhering to industry or internal company standards (AppSecure.nrw, 2021). Thus, knowledge sharing within the team is essential and can be promoted by dedicated roles such as an SC (Bodin & Golberg, 2021), as previously described. An evaluation should ascertain whether the team comprehends the security best practices germane to their product and if efficient knowledge-sharing mechanisms are established.

Training: Enhancing security competency of team members reduces security risks (Bartsch, 2011; van der Heijden et al., 2018) and is essential for organizations (Synopsys, 2021). On-demand self-learning resources are particularly advantageous in LSAD environments (Poth et al., 2020). Hands-on security training, specifically tailored for developers, offers substantial value, for instance, by illustrating common software vulnerabilities (Bodin & Golberg, 2021). Therefore, measurement should focus on whether the team regularly improves its capability on security-related subjects.

Collaboration: Beyond internal teamwork, collaboration with external stakeholders is crucial for security in LSAD (Nägele et al., 2022). Agile teams may lack expertise to address security issues independently. In practice, security engineers (SEs) or (security) architects support on request (Britto et al., 2016), and agile teams collaborate with information security officers (ISOs) to discuss risks, potential countermeasures as well as their implementation (van der Heijden et al., 2018). Thus, the team maturity assessment should incorporate factors such as the quality and frequency of interactions with external stakeholders like SEs or ISOs.

5.2. Criteria associated with the SDLC

Activities: Conducting security activities is indispensable throughout the SDLC for enhancing software security. They bolster a software product's security posture and augment security knowledge and awareness (Nägele et al., 2022). Hence, the selection, quality, and frequency of security activities performed by a development team can serve as an indicator of their maturity level. The more a team refines its proficiency in individual security activities, the greater its overall security posture is enhanced. The simplest maturity assessment based on this criterion may examine whether a team routinely engages in suitable activities. These could include, e.g., threat modeling, penetration tests, the application of static (SAST) and dynamic application security testing (DAST) tools during development, or security code reviews prior to deployments. More advanced assessments could also examine the quality of those activities.

Development: Mature teams excel at preemptively preventing or fixing vulnerabilities during development (Pagel, 2020). This early remediation is beneficial because fixing defects during testing or maintenance is significantly more expensive than in the development phase (Dawson et al., 2010). An important mechanism to detect security issues as early as possible is the usage of quality gates, which could be defined as a minimum requirement to achieve higher maturity (AppSecure.nrw, 2021). Our model recommends encouraging automated capabilities during the team's journey to increase their security maturity. Instead of manual reviews, automated quality gates, e.g., through SAST and DAST tools, should be used whenever the criticality of the release allows it. Organizations could also evaluate if their teams learn from past vulnerabilities and incorporate their detection into the development process.

Documentation: Writing sufficient documentation without impeding agility presents a considerable challenge in agile development (Beznosov & Kruchten, 2004). Proper documentation is crucial for security compliance and enhances transparency, fosters understanding, and aids in maintaining and further developing software (Alsaqaf et al., 2017). In addition, specific security activities, such as threat modeling, necessitate a certain level of documentation, for example, architectural diagrams. Given the significance of documentation, we incorporate it as a criterion for team maturity assessment. However, we advise evaluating not merely the documentation quality but also the level of effort and its integration into iterative workflows. Security documentation should maintain a delicate balance between a sufficient level of detail, consistent structure, and low overhead.

Product: The quality of the outcomes produced by teams can serve as a measure of maturity. Audits conducted by external organizations can provide an objective evaluation of a software product (Bartsch, 2011). Such audits may also indirectly motivate developers to write secure code to avoid potential embarrassment (Bodin & Golberg, 2021). Examples include penetration testing, bug bounty programs, or security code and architecture reviews (Synopsys, 2021).

Alternatively, internal resources could be deployed for the same purpose. Product assessments offer the benefit of objectively determining whether the team's performance aligns with the organization's quality benchmarks, facilitating comparative analysis across teams (Bartsch, 2011). We propose that the quality of a team's product, as determined by non-team members through reviews, be used as an evaluation criterion for their security maturity. We further recommend evaluating the team's response to feedback, such as how identified security risks or vulnerabilities are addressed.

Responsibility: The responsibility for security in development teams is often unclear (van der Heijden et al., 2018), making it crucial yet challenging to precisely articulate security requirements, integrate them into agile workflows, and distribute responsibilities. Issues may arise when central security teams issue vague or overly prescriptive requirements, worsened by interdependencies and shared accountability in scaled environments, or conflicting requirements from various SEs or ISOs (Alsaqaf et al., 2017). Hence, it is essential to evaluate if a team regularly identifies security requirements, integrates them into agile methods, and clearly assigns responsibilities.

6. Team security maturity model (TSMM)

The TSMM is an exemplary model to determine the maturity of a development team in developing secure and security-compliant applications, based on the criteria previously presented in Section 5.

In the following, we first outline the structure of our model and then explain its composition. More details can be found in our supplementary material (Nägele et al., 2023).

6.1. Overall structure

To diagnose and enhance the state of development teams with minimal complexity, we decided to construct a maturity grid, aligning with the inherent objective of such models (Maier et al., 2012). Its assessment mode, featuring textual description to determine maturity levels, lends practicality to our model by enabling teams to understand the maturity levels for each topic more easily, thereby allowing better self-classification of their maturity. We chose four maturity levels for each topic to avoid an "escape category" that may emerge from an odd number of response options because it is perceived as a mean (Porst, 2011).

We propose three data sources to optimize determining the maturity score: Self-assessments by development teams, assessments by roles external to the team, and systems from which data can be extracted (semi-)automatically.

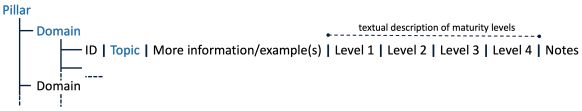

Figure 2. Outline of the TSMM structure

Following Poth et al. (2021), our TSMM is structured into pillars, domains, and topics, as shown in Figure 2. The TSMM contains three *pillars*, corresponding to the data sources, with each pillar subdivided into *domains* for grouping similar content, and further divided into *topics* featuring precise statements for classifying maturity. We implemented the TSMM in two ways: In Excel, as a simple and universally applicable solution, and as a web application prototype.

6.2. Self-assessment

We propose that teams carry out the bulk of the TSMM assessment themselves in a decentralized manner to bolster ownership and distribute effort. Our aim is to concentrate on pivotal elements and streamline the self-assessment process to enhance acceptance. Subsequently, we briefly delineate each self-assessable domain.

Table 1. Excerpt of the TSMM knowledge domain

ID	Topic
K1	We understand the significance of security in the context of our product.
K2	We are aware of and adhere to internal and external standards relevant to our product.
K3	We facilitate knowledge-sharing sessions among team members.
K4	We identify and plan to rectify security knowledge gaps.
K5	We are aware of and use standardized components for secure development.

Knowledge: This domain primarily evaluates a team's security (compliance) knowledge. We present an exemplary excerpt of this domain from the TSMM in Table 1 for illustration purposes. Topic K1 encourages a profound understanding of the significance of security in relation to their product, which is designed to instill intrinsic motivation. Our interview study disclosed that many experts witnessed security standards and guidelines being viewed as superfluous burdens due to a lack of comprehensive understanding of their core purpose. Furthermore, we strive to nurture the exchange of security knowledge within the team through topic K3. This aims

to mitigate risks associated with knowledge siloing, such as potential loss due to sickness or departure.

Activities: The effective execution of security activities enhances product security and compliance. While the TSMM does not prescribe activities, it encourages teams to make informed decisions to choose activities best suited to their needs.

Documentation: Maintaining adequate documentation is critical for (security) compliance and enables certain security activities such as risk analysis. Thus, the TSMM topics encourage teams to create, maintain, and review security documentation while emphasizing the need for minimal overhead.

Build and deployment: This domain focuses on pre-release security, integrating the criteria *activities*, *development*, and *responsibility*. Emphasizing quality gates to uphold release security, we underscore regular component maintenance, patching, and backup creation. In addition, the topics encourage automation and low-effort security testing, urging a reproducible build process with integrated security measures. Maturity levels range from non-existent to a robust DevSecOps pipeline. In addition, a topic assesses the consideration of security throughout the SDLC for early vulnerability detection and remediation.

Organization: Mainly derived from the criteria *composition* and *collaboration*, the topics of the organization domain encourage teams that seek greater autonomy to take on defined security responsibilities. At least one developer should possess advanced security knowledge and guide the team, such as by serving as an SC or similar role. In addition, teams are encouraged to identify and consult security experts outside of the team, particularly in the context of LSAD. The model also addresses the need for defined procedures to resolve conflicting security requirements between teams and for handovers of security responsibilities, a key insight from a workshop.

6.3. External assessment

The external assessment pillar includes the two domains audit and culture and facilitates evaluation by individuals external to the team, aiming to validate and expand the self-assessments. Typical roles in an LSAD environment that could conduct such an assessment are SEs who collaborate with multiple agile teams.

Audit: The audit domain mainly integrates the *product* criterion by assessing the security maturity of the produced software products of a team. For this topic, it is constructive to use a security expert who knows the required standards and regulations that apply to the product to verify. In addition, the external reviewers could also assess the development processes, security automation, and quality gates.

Culture: The TSMM envisions external reviews not solely for technical maturity. External assessments might also be beneficial to evaluate the criteria of *training*, *composition*, and *collaboration*. The derived topics encourage teams to cultivate a culture of continuous improvement to achieve enhanced product quality. An external perspective could offer valuable insights into improvement paths. Regularly collaborating security experts may be utilized to assess the team's security maturity, drawing from their interactions with the team. Furthermore, the maturity calculations also consider participation in security communities.

6.4. Automated assessment

The automated pillar uses data from security tools, necessitating customization to each organization. Organizations must identify data sources, develop strategies for data extraction, transformation, and delivery, and select and evaluate the compatibility of performance metrics with the TSMM's four customizable maturity levels. A web application implementation of the TSMM offers flexible data integration options, while an Excel implementation, though more limited, provides options like SQL connectors. Besides the primary testing domain, the TSMM also features an auxiliary domain to outline automation opportunities.

Testing: The testing domain is based on the *activities* and *product* criteria. We propose that SAST and DAST tools are used as data sources, as well as results of manual penetration tests or bug bounty programs. We have formulated corresponding topics and suggest using suitable performance indicators, such as the amount and criticality of security findings, response times to rectify findings (aggregated by criticality), and amount and criticality of security incidents.

Auxiliary metrics: Automated metrics could extend beyond security testing tools and consider any data potentially indicative of a team's maturity level. Our topics encompass examples like the inclusion of security requirements or risk-labeled tickets in software project management tools, and the team's engagement and progress in security training tools. However, the chosen metrics should be non-intrusive, and the TSMM encourages teams to actively participate in selecting metrics and defining maturity levels, e.g., through decentralized security communities.

7. Evaluation

Generally, experts found the tripartite data sources for maturity score computation beneficial, with

particular approval for the automated pillar due to its facilitation of continuous assessments devoid of manual intervention. They deemed the TSMM content sensible and the grid-style nature very useful, albeit requiring customization to fit the context of each respective organization. Experts stressed that the defined topics and corresponding maturity levels are contingent upon numerous factors, such as employed technology, centrally provided tools and infrastructure, and the resulting responsibilities of individual development teams. Highlighted examples include management of backups for application infrastructure and their data, logging and monitoring practices, utilization and maintenance of third-party tools and libraries, and provision of security testing tools such as static code analysis and vulnerability scanners by central teams.

Experts underscored that the maturity model's topics should not excessively inhibit the autonomy of development teams. For instance, rather than mandating specific techniques for identifying and addressing common vulnerabilities, the assessment could ascertain the presence of such measures, not necessarily their exact form. Discussions also centered on whether team assessment should be primarily based on team capabilities or also incorporate their output, meaning whether an assessment of their developed software products should impact the team maturity score. A majority of experts favored the inclusion of product-specific aspects.

The external assessment pillar was valued for offering a more objective perspective, validating self-assessments, and revealing potential oversights. However, conflicts of interest were noted when team-external roles, like in-house security engineers, were funded by the team's budget. This situation was summarized by an expert as "you do not bite the hand that feeds you", indicating that while security engineers are well-suited for assessments, they may evaluate less critically if financed by the team they are assessing.

Several respondents highlighted the automated pillar's significance, as it offers continuous monitoring without the potentially lengthy gaps of self- or third-party assessments. One expert suggested considering whether teams should be able to override results, such as in cases of false positives, to maintain the relevance of maturity scores and team motivation. However, such overrides should be flagged for transparency. Finally, one interviewee stated that "it is essential to remember that you should not use a sledgehammer to crack a nut", stressing that organizations need to ensure that the scope of the assessment is appropriate and not overly burdensome.

8. Discussion

8.1. Key findings

Our research yielded five key findings which we summarize in the following.

First, the ten proposed team security maturity criteria provide a holistic foundation for creating new maturity models to evaluate the security competency of agile teams. Organizations could also utilize these criteria to assess the completeness of existing models. The expert interviewees concurred with the significance of these factors and did not identify any overlooked critical aspects, though they contributed valuable detailed insights. However, we acknowledge the potential existence of other significant factors not yet identified.

Second, introducing team maturity levels can ameliorate the tension between autonomy and control in LSAD. By fostering transparency and motivating teams to bolster their proficiency in creating secure and compliant applications, central security governance teams can allocate their finite resources more judiciously, e.g., by prioritizing the support of low-maturity teams. Teams with a higher level of maturity, on the other hand, may work more autonomously, thereby better aligning with agile methods. Therefore, the TSMM requires a sustainable integration with security governance, functioning more as a facilitator than a traditional controller. Despite potential initial resistance from central governance and security teams due to concerns of diminished influence and control, we expect that the overall security posture and value creation significantly profit from empowering development teams to improve their security maturity and take greater responsibility. As development teams are closest to their own products, they are best positioned to secure them, given adequate security capabilities. As a result, increasing team maturity also mitigates some of the unique security challenges of LSAD, for example, the alignment of security objectives in distributed settings (van der Heijden et al., 2018). More mature teams require less security coordination and quality assurance.

Third, our evaluations demonstrate a preference among experts for a mixed source approach in calculating team maturity scores, integrating self- and external assessments, and (semi-)automated metrics. While each source may be biased in isolation, their combination yields a holistic perspective.

Fourth, the TSMM offers considerable transparency and feedback, providing insights into an organization's security posture and guiding teams to improve by identifying potential weak spots and areas requiring training. By analyzing team maturity profiles,

organizations can optimize security measures, whether by replacing outdated controls or extrapolating organization-wide actions from teams producing the most secure applications.

Finally, the exact configuration of assessment types, maturity levels, and content of the model is closely linked to the organizational structure, prevailing product risks, and predominant technologies utilized by development teams. Consequently, the main contribution of our TSMM lies less in the specific details of the model and more in its overarching idea and structure. Our primary intent in presenting the TSMM is to inspire organizations to harness the potential of team security maturity and adapt their governance procedures and empower agile teams without compromising security.

The TSMM fills a gap by evaluating maturity of capabilities with a grid-style approach rather than providing a checklist of required practices. It employs descriptive text for classifying maturity levels, aiding self-assessments. This concept could be applied to other concerns of product quality besides security.

8.2. Limitations

In order to adhere to curtail potential research limitations, our study closely followed the empirical standards for software engineering research (ACM SIGSOFT, 2023) during both the interviews and systematic review. The interviews served to evaluate our results. However, given the restrictive time frame of an interview, an exhaustive exploration of each topic along with the practical application of the model within an LSAD environment—particularly observing its impact on factors such as the overall security maturity of development teams—remained beyond the scope of this study. This represents a limitation of our present research, and the exploration of these aspects forms part of our future research agenda.

To improve reliability, all interviews were recorded and transcribed. For the analysis, we conducted systematic, reproducible content analysis and classification as described by Kuckartz (2016). To ensure construct validity (Runeson & Höst, 2009) in our interviews, we employed semi-structured questionnaires, which were first tested for comprehensibility, and ambiguities were clarified directly through dialogues with the interviewees. However, given the conversational nature of the semi-structured approach, potential deviations and imprecisions were inherently difficult to eliminate. Lastly, to scrutinize and offset threats to the validity of our designed artifact ensuing from our DSR process, we utilized the guidelines proffered by Lukyanenko et al. (2014).

9. Conclusion

The rise of scaled agile methods and the growing importance of security create tension between autonomy and control in LSAD. To ease this conflict, we encourage using security maturity scores for development teams. We used DSR to address our RQ on designing a team security maturity model, involving an SLR, expert interviews, and workshops to achieve rigor and practical relevance.

We found ten key criteria from the literature to assess a team's ability to develop secure applications, which we used to create and evaluate the maturity model through expert interviews and workshops. Since our evaluation has been limited to expert interviews, future studies could apply and analyze the TSMM in LSAD settings, examining its effectiveness in measuring team capability and implications on the autonomy-control tension during actual use.

References

ACM SIGSOFT. (2023). *Empirical standards for conducting and evaluating research in software engineering*. Retrieved June 14, 2023, from github.com/acmsigsoft/EmpiricalStandards

Alsaqaf, W., Daneva, M., & Wieringa, R. (2017). Quality requirements in large-scale distributed agile projects – a systematic literature review. *Lecture Notes in Computer Science*, *10153*, 219–234.

AppSecure.nrw. (2021). *Security-belts*. Retrieved June 14, 2023, from github.com/AppSecure-nrw/security-belts

Bartsch, S. (2011). Practitioners' perspectives on security in agile development. *2011 Sixth International Conference on Availability, Reliability and Security*, 479–484.

Becker, J., Knackstedt, R., & Pöppelbuß, J. (2009). Developing maturity models for it management. *Business & Information Systems Engineering*, *1*(3), 213–222.

Bell, L., Brunton-Spall, M., Smith, R., & Bird, J. (2017). *Agile application security: Enabling security in a continuous delivery pipeline*. O'Reilly.

Beznosov, K., & Kruchten, P. (2004). Towards agile security assurance. *Proceedings of the 2004 workshop on New security paradigms*, 47–54.

Bishop, D., & Rowland, P. (2019). Agile and secure software development: An unfinished story.

Bodin, N., & Golberg, H. K. B. (2021). *Software security culture in development teams: An empirical study*. NTNU.

Britto, R., Smite, D., & Damm, L.-O. (2016). Software architects in large-scale distributed projects: An ericsson case study. *IEEE Software*, *33*(6), 48–55.

Dawson, M., Burrell, D., Rahim, E., & Brewster, S. (2010). Integrating software assurance into the software development life cycle (sdlc). *Journal of Information Systems Technology and Planning*, *3*, 49–53.

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, *119*, 87–108.

Döring, N., Bortz, J., Pöschl, S., Werner, C. S., Schermelleh-Engel, K., Gerhard, C., & Gäde, J. C. (2016). *Forschungsmethoden und evaluation in den sozial- und humanwissenschaften*. Springer.

Horlach, B., Böhmann, T., Schirmer, I., & Drews, P. (2018). It governance in scaling agile frameworks. In *Mkwi 2018*.

Jaatun, M. G., & Soares Cruzes, D. (2021). Care and feeding of your security champion. *Int. Conf. on Cyber Situational Awareness, Data Analytics and Assessment*, 1–7.

Julisch, K. (2008). Security compliance. *Proceedings of the 2008 workshop on New security paradigms*, 71.

Kuckartz, U. (2016). *Qualitative inhaltsanalyse. methoden, praxis, computerunterstützung*. Beltz.

Lukyanenko, R., Evermann, J., & Parsons, J. (2014). Instantiation validity in is design research. Springer.

Maier, A. M., Moultrie, J., & Clarkson, P. J. (2012). Assessing organizational capabilities: Reviewing and guiding the development of maturity grids. *IEEE Transactions on Engineering Management*, *59*(1), 138–159.

Nägele, S., Watzelt, J.-P., & Matthes, F. (2022). Investigating the current state of security in large-scale agile development. *23rd Int. Conf. on Agile Software Development (XP)*, 203–219.

Nägele, S., Watzelt, J.-P., & Matthes, F. (2023). *Supplementary material*. Retrieved September 3, 2023, from https://doi.org/10.6084/m9.figshare.c.6819243.v1

OWASP Foundation. (2022). *Owasp samm v2*. Retrieved June 14, 2023, from owaspsamm.org/

Pagel, T. (2020). *Owasp devsecops maturity model*. Retrieved June 14, 2023, from dsomm.timo-pagel.de/

Peffers, K., Tuunanen, T., Rothenberger, M., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, *24*, 45–77.

Porst, R. (2011). *Fragebogen: Ein Arbeitsbuch*. VS Verlag für Sozialwissenschaften.

Poth, A., Kottke, M., Mahr, T., & Riel, A. (2021). Teamwork quality in technology-driven product teams in large-scale agile organizations. *Journal of Software: Evolution and Process*, e2388.

Poth, A., Kottke, M., & Riel, A. (2020). Scaling agile on large enterprise level with self-service kits to support autonomous teams. *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, 731–737.

Poth, A., Kottke, M., & Riel, A. (2021). Measuring teamwork quality in large-scale agile organisations evaluation and integration of the atwq approach. *IET Software*, *15*, 443–452.

Rindell, K., Ruohonen, J., Holvitie, J., Hyrynsalmi, S., & Leppänen, V. (2021). Security in agile software development: A practitioner survey. *Information and Software Technology*, *131*, 106488.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131–164.

Synopsys. (2021). *Building security in maturity model 12 — bsimm*. Retrieved March 1, 2022, from bsimm . com / content / dam / bsimm / reports / bsimm12-foundations.pdf

Uludağ, Ö., Putta, A., Paasivaara, M., & Matthes, F. (2021). Evolution of the agile scaling frameworks. *22nd Int. Conf. on Agile Software Development (XP)*, 123–139.

van der Heijden, A., Broasca, C., & Serebrenik, A. (2018). An empirical perspective on security challenges in large-scale agile software development. *12th ACM/IEEE Int. Symposium on Empirical Software Engineering and Measurement*.

von Solms, R. (1998). Information security management: The code of practice. *Information Management & Computer Security*, *6*(5), 224–225.

Webster, J., & Watson, R. T. (2002). Analyzing the past to prepare for the future: Writing a literature review.

Wendler, R. (2012). The maturity of maturity model research: A systematic mapping study. *Information and Software Technology*, *54*(12), 1317–1339.

APPENDIX B

License Agreements for Embedded Publications

The following pages contain the license agreements for each publication, in order of the appearance of their respective publication in Appendix A (**P1** to **P5**).

Licence to Publish
Proceedings Papers

SPRINGER NATURE

Licensee	Springer Nature Switzerland AG	(the 'Licensee')
Title of the Proceedings Volume/Edited Book or Conference Name:	XP 2022 conference	(the 'Volume')
Volume Editor(s) Name(s):	V. Stray, K.J. Stol, M. Paasivaara, P. Kruchten	
Proposed Title of the Contribution:	Investigating the Current State of Security in Large-Scale Agile Development	(the 'Contribution')
Series: The Contribution may be published in the following series	A Springer Nature Computer Science book series (CCIS, LNAI, LNBI, LNBIP or LNCS)	
Author(s) Full Name(s):	Sascha Nägele, Jan-Philipp Watzelt, Florian Matthes	(the 'Author')

When Author is more than one person the expression "Author" as used in this Agreement will apply collectively unless otherwise indicated.

Corresponding Author Name:	Sascha Nägele	
Instructions for Authors	https://resource-cms.springernature.com/springer-cms/rest/v1/content/19242230/data/	(the 'Instructions for Authors')

Licence Applicable to the Contribution:

Creative Commons licence CC BY: This licence allows readers to copy, distribute and transmit the Contribution as long as it is attributed back to the author. Readers are permitted to alter, transform or build upon the Contribution, and to use the Contribution for commercial purposes. Please read the full licence for further details at - http://creativecommons.org/licenses/by/4.0/

Subject to editorial acceptance of the Contribution, it will be published under the Creative Commons licence shown above.

1 Grant of Rights

a) For good and valuable consideration, the Author hereby grants to the Licensee the perpetual, non-exclusive, irrevocable, world-wide, assignable, sublicensable and unlimited right to: publish, reproduce, copy, distribute, communicate, display publicly, sell, rent and/or otherwise make available the contribution identified above, including any supplementary information and graphic elements therein (e.g. illustrations, charts, moving images) (the 'Contribution') in any language, in any versions or editions in any and all forms and/or media of expression (including without limitation in connection with any and all end-user devices), whether now known or developed in the future. Without limitation, the above grant includes: (i) the right to edit, alter, adapt, adjust and prepare derivative works; (ii) all advertising and marketing rights including without limitation in relation to social media; (iii) rights for any training, educational and/or instructional purposes; (iv) the right to add and/or remove links or combinations with other media/works; and (v) the right to create, use and/or license and/or sublicense content data or metadata of any kind in relation to the Contribution (including abstracts and summaries) without restriction. The

above rights are granted in relation to the Contribution as a whole or any part and with or in relation to any other works.

b) Without limiting the rights granted above, Licensee is granted the rights to use the Contribution for the purposes of analysis, testing, and development of publishing- and research-related workflows, systems, products, projects, and services; to confidentially share the Contribution with select third parties to do the same; and to retain and store the Contribution and any associated correspondence/files/forms to maintain the historical record, and to facilitate research integrity investigations. The grant of rights set forth in this clause (b) is irrevocable.

c) If the Licensee elects not to publish the Contribution for any reason, all publishing rights under this Agreement as set forth in clause 1a above will revert to the Author.

2 Copyright

Ownership of copyright in the Contribution will be vested in the name of the Author. When reproducing the Contribution or extracts from it, the Author will acknowledge and reference first publication in the Volume.

3 Use of Contribution Versions

a) For purposes of this Agreement: (i) references to the "Contribution" include all versions of the Contribution; (ii) "Submitted Manuscript" means the version of the Contribution as first submitted by the Author prior to peer review; (iii) "Accepted Manuscript" means the version of the Contribution accepted for publication, but prior to copy-editing and typesetting; and (iv) "Version of Record" means the version of the Contribution published by the Licensee, after copy-editing and typesetting. Rights to all versions of the Manuscript are granted on a non-exclusive basis.

b) The Author may make the Submitted Manuscript available at any time and under any terms (including, but not limited to, under a CC BY licence), at the Author's discretion. Once the Contribution has been published, the Author will include an acknowledgement and provide a link to the Version of Record on the publisher's website: "This preprint has not undergone peer review (when applicable) or any post-submission improvements or corrections. The Version of Record of this contribution is published in [insert volume title], and is available online at https://doi.org/[insert DOI]".

c) Immediately after acceptance the Author may deposit the Accepted Manuscript to any location, and under any terms (including, but not limited to, under a CC BY licence), provided it is not made publicly available until after publication. The Author will include an acknowledgement in the Accepted Manuscript, together with a link to the Version of Record on the publisher's website: "This version of the contribution has been accepted for publication, after peer review (when applicable) but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/[insert DOI]".
Any use of the Accepted Manuscript not expressly permitted under this subclause (c) is subject to the Licensee's prior consent.

4 Warranties & Representations

Author warrants and represents that:

a)

i. the Author is the sole copyright owner or has been authorised by any additional copyright owner(s) to grant the rights defined in clause 1,

ii. the Contribution does not infringe any intellectual property rights (including without limitation copyright, database rights or trade mark rights) or other third party rights and no licence from or payments to a third party are required to publish the Contribution,

iii. the Contribution has not been previously published or licensed, nor has the Author committed to licensing any version of the Contribution under a licence inconsistent with the terms of this Agreement,

iv. if the Contribution contains materials from other sources (e.g. illustrations, tables, text quotations), Author has obtained written permissions to the extent necessary from the copyright holder(s), to license to the Licensee the same rights as set out in clause 1 and has cited any such materials correctly;

b) all of the facts contained in the Contribution are according to the current body of research true and accurate;

c) nothing in the Contribution is obscene, defamatory, violates any right of privacy or publicity, infringes any other human, personal or other rights of any person or entity or is otherwise unlawful and that informed consent to publish has been obtained for any research participants;

d) nothing in the Contribution infringes any duty of confidentiality owed to any third party or violates any contract, express or implied, of the Author;

e) all institutional, governmental, and/or other approvals which may be required in connection with the research reflected in the Contribution have been obtained and continue in effect;

f) all statements and declarations made by the Author in connection with the Contribution are true and correct;

g) the signatory who has signed this Agreement has full right, power and authority to enter into this Agreement on behalf of all of the Authors; and

h) the Author complies in full with: i. all instructions and policies in the Instructions for Authors, ii. the Licensee's ethics rules (available at https://www.springernature.com/gp/authors/book-authors-code-of-conduct), as may be updated by the Licensee at any time in its sole discretion.

5 Cooperation

a) The Author will cooperate fully with the Licensee in relation to any legal action that might

arise from the publication of the Contribution, and the Author will give the Licensee access at reasonable times to any relevant accounts, documents and records within the power or control of the Author. The Author agrees that any Licensee affiliate through which the Licensee exercises any rights or performs any obligations under this Agreement is intended to have the benefit of and will have the right to enforce the terms of this Agreement.

b) Author authorises the Licensee to take such steps as it considers necessary at its own expense in the Author's name(s) and on their behalf if the Licensee believes that a third party is infringing or is likely to infringe copyright in the Contribution including but not limited to initiating legal proceedings.

6 Author List

Changes of authorship, including, but not limited to, changes in the corresponding author or the sequence of authors, are not permitted after acceptance of a manuscript.

7 Post Publication Actions

The Author agrees that the Licensee may remove or retract the Contribution or publish a correction or other notice in relation to the Contribution if the Licensee determines that such actions are appropriate from an editorial, research integrity, or legal perspective.

8 Controlling Terms

The terms of this Agreement will supersede any other terms that the Author or any third party may assert apply to any version of the Contribution.

9 Governing Law

This Agreement shall be governed by, and shall be construed in accordance with, the laws of Switzerland. The courts of Zug, Switzerland shall have the exclusive jurisdiction.

Signed for and on behalf of the Author	Print Name:	Date:
[redacted]	Sascha Nägele	08.04.2022

Address: [redacted]	
Email: [redacted]	

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

The Current State of Security Governance and Compliance in Large-Scale Agile Development: A Systematic Literature Review and Interview Study

Sascha Nägele, Nathalie Schenk, Florian Matthes

2023 IEEE 25th Conference on Business Informatics (CBI)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.

2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.

3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.

4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.

5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.

6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO NOT wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Sascha Nägele — **Signature**

21-06-2023 — **Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

202

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE

PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the IEEE work with a Digital Object Identifier (DOI) or link to the article abstract in IEEE Xplore, or (2) the accepted version only (not the IEEE-published version), including the IEEE copyright notice and full citation, with a link to the final, published article in IEEE Xplore.

Questions about the submission of the form or manuscript must be sent to the publication's editor.
Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966

ACM Publishing License and Audio/Video Release

Title of the Work: Adoption of Information Security Practices in Large-Scale Agile Software Development: A Case Study in the Finance Industry
Submission ID:6447186

Author/Presenter(s): Sascha Nägele·Technical University Munich;Lorena Korn·Technical University Munich;Florian Matthes·Technical University Munich

Type of material:full paper

Publication and/or Conference Name: The 18th International Conference on Availability, Reliability and Security Proceedings

1. Glossary

2. Grant of Rights

(a) Owner hereby grants to ACM an exclusive, worldwide, royalty-free, perpetual, irrevocable, transferable and sublicenseable license to publish, reproduce and distribute all or any part of the Work in any and all forms of media, now or hereafter known, including in the above publication and in the ACM Digital Library, and to authorize third parties to do the same.

(b) In connection with software and "Artistic Images and "Auxiliary Materials, Owner grants ACM non-exclusive permission to publish, reproduce and distribute in any and all forms of media, now or hereafter known, including in the above publication and in the ACM Digital Library.

(c) In connection with any "Minor Revision", that is, a derivative work containing less than twenty-five percent (25%) of new substantive material, Owner hereby grants to ACM all rights in the Minor Revision that Owner grants to ACM with respect to the Work, and all terms of this Agreement shall apply to the Minor Revision.
(d) If your paper is withdrawn before it is published in the ACM Digital Library, the rights revert back to the author(s).

☑ A. Grant of Rights. I grant the rights and agree to the terms described above.

☐ B. Declaration for Government Work. I am an employee of the national government of my country/region and my Government claims rights to this work, or it is not copyrightable (Government work is classified as Public Domain in U.S. only)

Are you a contractor of your National Government? ○ Yes ◉ No

Are any of the co-authors, employees or contractors of a National Government? ○ Yes ◉ No

---

3. Reserved Rights and Permitted Uses.

(a) All rights and permissions the author has not granted to ACM in Paragraph 2 are

reserved to the Owner, including without limitation the ownership of the copyright of the Work and all other proprietary rights such as patent or trademark rights.

(b) Furthermore, notwithstanding the exclusive rights the Owner has granted to ACM in Paragraph 2(a), Owner shall have the right to do the following:

(i) Reuse any portion of the Work, without fee, in any future works written or edited by the Author, including books, lectures and presentations in any and all media.

(ii) Create a "Major Revision" which is wholly owned by the author

(iii) Post the Accepted Version of the Work on (1) the Author's home page, (2) the Owner's institutional repository, (3) any repository legally mandated by an agency funding the research on which the Work is based, and (4) any non-commercial repository or aggregation that does not duplicate ACM tables of contents, i.e., whose patterns of links do not substantially duplicate an ACM-copyrighted volume or issue. Non-commercial repositories are here understood as repositories owned by non-profit organizations that do not charge a fee for accessing deposited articles and that do not sell advertising or otherwise profit from serving articles.

(iv) Post an "Author-Izer" link enabling free downloads of the Version of Record in the ACM Digital Library on (1) the Author's home page or (2) the Owner's institutional repository;

(v) Prior to commencement of the ACM peer review process, post the version of the Work as submitted to ACM ("Submitted Version" or any earlier versions) to non-peer reviewed servers;

(vi) Make free distributions of the final published Version of Record internally to the Owner's employees, if applicable;

(vii) Make free distributions of the published Version of Record for Classroom and Personal Use;

(viii) Bundle the Work in any of Owner's software distributions; and

(ix) Use any Auxiliary Material independent from the Work.

When preparing your paper for submission using the ACM TeX templates, the rights and permissions information and the bibliographic strip must appear on the lower left hand portion of the first page.

The new ACM Consolidated TeX template Version 1.3 and above automatically creates and positions these text blocks for you based on the code snippet which is system-generated based on your rights management choice and this particular conference. When creating your document, please make sure that you are only using TAPS accepted packages. (If you would like to use a package not on the list, please send suggestions to acmtexsupport@aptaracorp.com RE: TAPS LaTeX Package evaluation.)

NOTE: For authors using the ACM Microsoft Word Master Article Template and Publication Workflow, The ACM Publishing System (TAPS) will add the rights statement to your papers for you. Please check with your conference contact for information regarding submitting your source file(s) for processing.

Please put the following LaTeX commands in the preamble of your document - i.e., before \begin{document}:

\copyrightyear{2023}
\acmYear{2023}
\setcopyright{acmlicensed}\acmConference[ARES 2023]{The 18th International Conference on Availability, Reliability and Security}{August 29-September 1, 2023}{Benevento, Italy}
\acmBooktitle{The 18th International Conference on Availability, Reliability and Security (ARES 2023), August 29-September 1, 2023, Benevento, Italy}
\acmPrice{15.00}
\acmDOI{10.1145/3600160.3600170}
\acmISBN{979-8-4007-0772-8/23/08}

NOTE: For authors using the ACM Microsoft Word Master Article Template and Publication Workflow, The ACM Publishing System (TAPS) will add the rights statement to your papers for you. Please check with your conference contact for information regarding submitting your source file(s) for processing.

If you are using the ACM Interim Microsoft Word template, or still using or older versions of the ACM SIGCHI template, you must copy and paste the following text block into your document as per the instructions provided with the templates you are using:

ARES 2023, August 29-September 1, 2023, Benevento, Italy

ACM ISBN 979-8-4007-0772-8/23/08...$15.00
https://doi.org/10.1145/3600160.3600170

NOTE: Make sure to include your article's DOI as part of the bibstrip data; DOIs will be registered and become active shortly after publication in the ACM Digital Library.

Once you have your camera ready copy ready, please send your source files and PDF to your event contact for processing.

---

4. ACM Citation and Digital Object Identifier.

(a) In connection with any use by the Owner of the Definitive Version, Owner shall include the ACM citation and ACM Digital Object Identifier (DOI).
(b) In connection with any use by the Owner of the Submitted Version (if accepted) or the Accepted Version or a Minor Revision, Owner shall use best efforts to display the ACM citation, along with a statement substantially similar to the following:

> "© [Owner] [Year]. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive version was published in {Source Publication}, https://doi.org/10.1145/{number}."

5. Livestreaming and Distribution

You are giving a presentation at the annual conference. This section of the rights form gives you the opportunity to grant or deny ACM the ability to make this presentation more widely seen, through (a) livestreaming of the presentation during the conference and/or (b) distributing the presentation after the conference in the ACM Digital Library, the "Conference Presentations" USB, and media outlets such as Vimeo and YouTube. It also provides you the opportunity to grant or deny our use of the presentation in promotional and marketing efforts after the conference.

Not all conference presentations are livestreamed; you will be notified in advance of the possibility of your presentation being livestreamed.

The permissions granted and/or denied here apply to all presentations of this material at the conference, including (but not limited to) the primary presentation and any program-specific "fast forward" presentations.

ACM's policy on the use of third-party material applies to your presentation as well as the documentation of your work; if you are using others' material in your presentation, including audio, you must identify that material on the ACM rights form and in the presentation where it is used, and secure permission to use the material where necessary.

Livestreaming.
I grant permission to ACM to livestream my presentation during the conference (a "livestream" is a synchronous distribution of the presentation to the public, separate from the presentation distributed to conference registrants).
(●) Yes
( ) No

Post-Conference Distribution.
I grant permission to ACM to distribute the recording of my presentation after the

conference as listed above.
(●) Yes
( ) No

6. Auxiliary Material

Do you have any Auxiliary Materials? ( ) Yes (●) No

7. Third Party Materials

In the event that any materials used in my presentation or Auxiliary Materials contain the work of third-party individuals or organizations (including copyrighted music or movie excerpts or anything not owned by me), I understand that it is my responsibility to secure any necessary permissions and/or licenses for print and/or digital publication, and cite or attach them below.

(●) We/I have not used third-party material.
( ) We/I have used third-party materials and have necessary permissions.

8. Artistic Images

If your paper includes images that were created for any purpose other than this paper and to which you or your employer claim copyright, you must complete Part IV and be sure to include a notice of copyright with each such image in the paper.
(●) We/I do not have any artistic images.
( ) We/I have any artistic images.

---

9. Representations, Warranties and Covenants

The undersigned hereby represents, warrants and covenants as follows:

(a) Owner is the sole owner or authorized agent of Owner(s) of the Work;

(b) The undersigned is authorized to enter into this Agreement and grant the rights included in this license to ACM;

(c) The Work is original and does not infringe the rights of any third party; all permissions for use of third-party materials consistent in scope and duration with the rights granted to ACM have been obtained, copies of such permissions have been provided to ACM, and the Work as submitted to ACM clearly and accurately indicates the credit to the proprietors of any such third-party materials (including any applicable copyright notice), or will be revised to indicate such credit;

(d) The Work has not been published except for informal postings on non-peer reviewed servers, and Owner covenants to use best efforts to place ACM DOI pointers on any such prior postings;

(e) The Auxiliary Materials, if any, contain no malicious code, virus, trojan horse or other software routines or hardware components designed to permit unauthorized access or to disable, erase or otherwise harm any computer systems or software; and

(f) The Artistic Images, if any, are clearly and accurately noted as such (including any applicable copyright notice) in the Submitted Version.

☑ I agree to the Representations, Warranties and Covenants.

---

10. Enforcement.

At ACM's expense, ACM shall have the right (but not the obligation) to defend and enforce the rights granted to ACM hereunder, including in connection with any instances of plagiarism brought to the attention of ACM. Owner shall notify ACM in writing as promptly as practicable upon becoming aware that any third party is infringing upon the rights granted to ACM, and shall reasonably cooperate with ACM in its defense or enforcement.

---

11. Governing Law

This Agreement shall be governed by, and construed in accordance with, the laws of the state of New York applicable to contracts entered into and to be fully performed therein.

Funding Agents

1. Bundesministerium für Bildung und Forschung award number(s):01IS17049

DATE: **06/24/2023** sent to sascha.naegele@tum.de at **05:06:25**

CONSENT TO PUBLISH and COPYRIGHT TRANSFER

For the mutual benefit and protection of Authors and Publishers, it is necessary that Authors provide formal written agreement to this Consent to Publish and Copyright Transfer ("Agreement") before publication of the Book. This Agreement ensures that the Publisher has the Author´s authorization to publish the Contribution in the event proceedings.

Event: **ICEIS 2024 - 26th International Conference on Enterprise Information Systems**.
Place/Date: **Angers, France; 28 - 30 April, 2024**.
Book Title: **Proceedings of the 26th International Conference on Enterprise Information Systems**.
Edited by: **Joaquim Filipe, Michał Śmiałek, Alexander Brodsky and Slimane Hammoudi**.
Publisher: **SCITEPRESS (Science and Technology Publications, Lda)**.

Paper number: **86**.
Contribution Title: **Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development**.

Author (**name** and address):
Sascha Nägele.
Germany.

It is herein agreed that:
The copyright to the Contribution identified above is transferred from the Author to the **"Science and Technology Publications, Lda"** (herein known as **"SCITEPRESS"**) during the full term of copyright. The copyright transfer covers the exclusive, sole, permanent, world-wide, transferable, sub licensable and unlimited right during the full term of copyright to reproduce, publish, transmit, archive, lease/lend, sell and distribute the Contribution or parts thereof individually or together with other works in any language, revision and version (digital and hard), including reprints, translations, photographic reproductions, microform, audiograms, videograms, electronic form (offline, online), or any other reproductions of similar nature, including publication in the aforementioned Book or any other book, as well as, the usage for advertising purposes. SCITEPRESS is also entitled to carry out editorial changes in the Contribution with the sole purpose of enhancing the overall organization and form of the Contribution. Notwithstanding the foregoing transfer of copyright, each Author, and each Author's employer, retains all proprietary rights other than copyright, such as patent rights, in any process, procedure or article of manufacture described in the Contribution. Each Author retains the rights and is licensed to use and publish the Contribution in his/her own web site and thesis, in the Author's employer's and its affiliated companies' web sites and to publish a substantially revised version (at least 30% new material) elsewhere, as long as it is clearly stated that the Contribution was presented at ICEIS 2024, a link to the ICEIS 2024 web site is made available and also lists the corresponding DOI number, if any. Prior versions of the Contribution published on non-commercial pre-print servers like ArXiv/CoRR and HAL can remain on these servers and/or can be updated with Author's accepted version. The final published version (in pdf) of the Contribution cannot be used for this purpose. The Creative Commons license CC BY-NC-ND applies to everyone that wishes to use the published version of the Contribution.
The Author warrants that the Contribution is original, except for such excerpts from copyrighted works as may be included with the permission of the copyright holder and author thereof, that it contains no libelous statements, and does not infringe on any copyright, trademark, patent, statutory right, or propriety right of others; and that Author will indemnify SCITEPRESS against any costs, expenses or damages for which SCITEPRESS may become liable as a result of any breach of this warranty. The Author signs for and accepts responsibility for releasing the Contribution for publication as specified in this Agreementt on behalf of any and all Co-Authors.
This Agreement shall be governed by, and shall be construed in accordance with, the laws of Portugal. The courts of Portugal shall have the exclusive jurisdiction.
In return for these rights granted in this Agreement, SCITEPRESS agrees as follows:

SCITEPRESS agrees to have the Contribution published, at its own cost and expense, in the event proceedings.
The undersigned Author hereby authorizes SCITEPRESS to publish the Contribution in the event proceedings. If the Contribution is not published in the event proceedings this Agreement will be null and void.

Date: **21 February, 2024**

Author´s Signature:

Sascha Nägele

Hawaii International Conference on System Sciences (HICSS)

Permanent URI for this community https://hdl.handle.net/10125/39610

The Hawaii International Conference on System Sciences (HICSS) has been known world-wide as one of the longest-standing scientific conferences. It provides a highly interactive working environment for top scholars from academia and the industry from over 40 countries to exchange ideas in various areas of information, computer, and system sciences.

HICSS features:

- Three days of research paper presentations and discussions in ten research tracks and themes that enable research on a rich mixture of computer-based applications and technologies
- A full day of Symposia, Workshops, and Tutorials
- Best Paper Awards in each track, which recognize superior research performance

Papers prior to HICSS-50 (2017) were published under the IEEE Copyrights. They can be accessed through the IEEE Computer Society Digital Library.

Papers from HICSS-50 (2017) onward are accessible through ScholarSpace. They were published with the Creative Commons licenses (CC-BY-NC-ND 4.0).

News

HICSS Contact Information

HICSS Conference Office University of Hawaii at Manoa
2404 Maile Way D307
Honolulu HI 96822
Tel: +1-808-956-3251
Email: hicss@hawaii.edu
Web site: https://hicss.hawaii.edu

HICSS Publishing Information

Series Title:
Proceedings of the Annual Hawaii International Conference on System Sciences

International Standard Serial Number (ISSN):
2572-6862 (online)

Browse

212

Sub-communities and Collections | By Issue Date | By Author | By Title | By Subject