INFORMATION SECURITY IN AGILE SOFTWARE DEVELOPMENT PROJECTS: A CRITICAL SUCCESS FACTOR PERSPECTIVE

Research paper

Nathan Newton, Victoria University of Wellington, Wellington, New Zealand, nathan.newton@myvuw.ac.nz

Craig Anslow, Victoria University of Wellington, Wellington, New Zealand, craig.anslow@vuw.ac.nz

Andreas Drechsler, Victoria University of Wellington, Wellington, New Zealand, andreas.drechsler@vuw.ac.nz

Abstract

The importance of information security in software development projects is long recognised, with many comprehensive standards and procedures in use to provide assurance of information security. The agile development paradigm conflicts with traditional security assurance by emphasising the delivery of functional requirements and a reduction in structured and linear development styles. Through a series of thirteen qualitative interviews, this study identifies practices that address this problem which have been successfully adopted by agile practitioners. The findings present four categories of practices – organisational, team, project, and technical – and twelve critical success factors that should be explicitly considered by practitioners to assure agile security. The critical success factors provide a foundation for practitioners to strategically identify and develop best practices to embed information security in agile development projects. The identified categories also highlight the importance of agile security practices centring around individuals and culture and contributes to the literature by providing a representation of agile security practices that encompasses a broad range of focal areas.

Keywords: information security; agile development; critical success factors

1 Introduction

Information systems (IS) are an increasingly centric component of an organisation's operational capabilities and competitive advantage (Chen et al., 2010; Peppard and Ward, 2004). However, as organisations become increasingly dependent upon these systems to create and sustain business value, a critical system failure or a compromise of sensitive business data holds significant organisational risk and consequences (Acar et al., 2017). The importance of information security and the potentially severe repercussions of an incident is evidenced in many recent cases. In 2018, the Baltimore emergency dispatch centre was rendered inoperable for 17 hours after succumbing to a ransomware attack (Rector, 2018), while in 2017, Equifax was involved in the unauthorised release of 146 million customers' personal data, after a third-party exploited a vulnerability in their systems (Bernard and Cowley, 2017). To contribute to comprehensive information security and mitigate the risk of such breaches, development teams need to adhere to rigid industry standards and structured processes (Sindre and Opdahl, 2005).

In response to shortfalls in traditional development methodologies, organisations and development teams are increasingly adopting the agile paradigm (Kropp et al., 2018; Licorish et al., 2016). Agile software development (ASD) methodologies emphasise adaptation to shifting requirements through flexible work practices and the rapid delivery of functional value to clients (Beck et al., 2001; Dingsøyr and Dybå, 2010; VersionOne and CollabNet, 2017). As a non-functional requirement, information security (InfoSec) is not typically considered to be a fundamental source of value to a client, and as such, is often treated as a lower priority than functional requirements in ASD, consequently resulting in a technical debt for security (Boehm and Turner, 2005; Chung and do Prado Leite, 2009; Curtis et al., 2012; Glinz, 2007). Furthermore, accepted industry standards for InfoSec mandate formal procedures that necessitate extensive documentation and rigorous testing. These approaches to assuring InfoSec contradict agile practices that are dependent on short iterations and rapid delivery of functionality (Bartsch, 2011; Hood, 2017).

As the security threats that organisations are exposed to increase in complexity and number, the perceived malalignment between ASD and InfoSec may leave information systems and organisations vulnerable to security threats, and at risk of both financial and reputational loss. Development cultures where InfoSec is considered an impediment to agile delivery and is at risk of being under-prioritised may be detrimental to the assurance of security in information systems. There is the need to identify new solutions for addressing InfoSec that better align with the values of the ASD paradigm to ensure that development teams can continue to effectively mitigate against the risk of a data breach or other InfoSec incidents while regularly delivering functional value to the client in a responsive manner.

Existing academic literature has identified this tension between InfoSec and ASD, and has made forays into recommending solutions, including security-oriented agile methodologies and techniques for documenting and prioritising non-functional requirements pertaining to InfoSec (Boström et al., 2006; Pohl and Hof, 2015). However, the majority of existing literature is conceptual, with few studies performing empirical research to understand the current state of InfoSec integration with ASD. Those few studies that perform empirical research are typically narrow in focus and describe only a limited range of approaches employed in practice.

To contribute towards closing the gap in existing literature, this research project investigates the current state of solutions for addressing InfoSec in ASD projects, identifying approaches used throughout the development lifecycle, and at different organisational levels. As the technical implementation of security counter-measures does not vary between ASD and traditional development methods, this study focuses primarily on project management and coordination practices for ensuring InfoSec. To achieve this research goal, a series of semi-structured interviews and subsequent qualitative analysis was conducted; the following research questions provided the focus for the study, leading to the identification of a categorised set of critical success factors for enabling InfoSec in ASD:

> **RQ1:** What solutions have been discussed in academic research for ensuring that the security needs of an information system are addressed appropriately in ASD?

RQ2: What solutions have practitioners adopted to ensure that the security needs of an information system are addressed appropriately in ASD?

RQ3: How do academic recommendations for addressing InfoSec needs in ASD differ from practice?

The remainder of this paper is structured as follows: Section 2 provides a foundation for three concepts that underpin this research; InfoSec, ASD, and critical success factors (CSF). Section 3 outlines the methodological approaches we used for our study. Section 4 contains a review of the existing literature relating to ASD and security. Section 5 presents the findings of the empirical research work. Section 6 discusses the implications, contributions, and limitations of this research. Section 7 draws a conclusion and outlines directions for further work.

2 Conceptual Foundations

This section provides an introduction to the three foundational concepts of this research. It describes the main concerns of InfoSec assurance, and what practices contemporary ASD entails. Critical success factors, a fundamental concept in our presented findings, are also defined and explained.

2.1 Information Security

Industry standards consider InfoSec to be concerned with the assurance of the confidentiality, integrity, and availability of an organisation's information assets (Andress, 2014; Bagiński and Rostański, 2011; Ellis, 2013; von Solms and van Niekerk, 2013). Unauthorised access to customers' personal information and commercially sensitive data must be mitigated against, as well as ensuring that this data remains accurate and untampered (Andress, 2014). Information systems must be able to operate uninterrupted, providing essential services even under atypical or potentially malicious operating circumstances to ensure business continuity (McGraw, 2006). Failure to sufficiently mitigate against a security incident can hold significant repercussions for an organisation, including disruptions to operations, financial costs, legal consequences, and reputational harm (Bellovin, 2015; Dynes et al., 2008).

The threat landscape faced by organisations is continually shifting as potential targets, attack vectors, and defence mechanisms evolve (Australian Computer Society, 2016), presenting a complex and varied risk landscape, necessitating a robust, yet adaptable, approach to addressing InfoSec during software development. Threats to an information system come in many forms, including malware, credential elevation, ransomware, digital vandalism, information leakage and obstruction of service (Choo, 2011; Collins, 2013; Kang et al., 2014).

As a widely recognised non-functional requirement, industry certifications and standards such as ISO 27001 or COBIT comprehensively address InfoSec (Höne and Eloff, 2002; Siponen, 2006; Siponen and Willison, 2009). These standards provide benchmarks for assessing the implementation of security mitigation techniques and provide frameworks for implementing InfoSec and benchmarks for assessing security risk mitigation, providing assurance that an organisation has taken reasonable precautions.

2.2 ASD Principles

ASD is not a methodology itself but rather is a set of guiding principles from which a multiplicity of methodologies have arisen (Elbanna and Sarker, 2016). The most commonly adopted of these methodologies is Scrum, though other popular methodologies include eXtreme Programming, Kanban, Lean, and hybrid approaches (Kropp et al., 2018; VersionOne and CollabNet, 2017). Each of these methodologies takes a different approach to development and project coordination yet share a commonality of adhering to the fundamental agile tenets.

Since the initial publication of the Agile Manifesto (Beck et al., 2001), the paradigm has become widely adopted throughout the software development industry (Dingsøyr et al., 2012; Licorish et al., 2016; Lindvall et al., 2002; VersionOne and CollabNet, 2017), as it addresses many challenges encountered in traditional 'Waterfall' style methodologies (Glass, 2001; Licorish et al., 2016; Petersen and Wohlin, 2009).

The agile paradigm proposes that the rapid delivery of functional value to a customer is essential to maintaining customer satisfaction (Beck et al., 2001). Delivering working software as early as possible provides the opportunity for project stakeholders to provide feedback on the product and allows for further refinement of customer requirements (Dingsøyr and Dybå, 2010; Dingsøyr et al., 2012; Petersen and Wohlin, 2009). By iteratively repeating this process with regular deliveries, the project team 'builds up' to a final product that is aligned with the stakeholder needs. To enable successful delivery in these conditions, open and regular communication through direct interactions within the team and stakeholders is prioritised over extensive documentation (Beck et al., 2001; Dingsøyr et al., 2012; Glass, 2001).

Teams should be comprised of motivated individuals, who together possess the full range of skills required for undertaking the project from conception to final delivery (Beck et al., 2001; Chau and Maurer, 2004). Management should empower the team, providing the necessary resources and autonomy to make decisions and self-organise, rather than adhering to traditional organisational hierarchies (Beck et al., 2001; Dingsøyr et al., 2012).

2.3 Critical Success Factors

CSFs are those areas of a business in which performance has a significant impact on an organisation's ability to succeed in attaining objectives, thereby supporting the competitiveness of a business and success of future endeavours (Leidecker and Bruno, 1984). Due to the importance of achieving highly in these areas, an organisation should provide specific and ongoing attention to ensure its capability of fulfilling these key performance areas (Boynton and Zmud, 1987).

Several key characteristics of CSFs have been proposed, allowing for accurate identification of factors essential to achieving organisational goals and objectives (Freund, 1988). CSFs are not an outcome of a process but should instead be expressed as activities within the process where high-performance is essential. Claiming that all activities and processes are critical detracts from the concept, by diminishing emphasis on those factors that are truly essential. CSFs do not exist within only a single hierarchical level of the organisation, but instead should be identified at organisational, unit, and functional levels, with each contributing to success in a different way. Finally, critical success factors are not specific to a single organisation but are generalisable to all organisations operating with a similar strategy in the same industry. Competitive advantage instead arises from each organisation's unique ability to fulfil these factors in a manner that capitalises on internally available strengths and resources.

3 Research Methodology

This section describes the research methodology we employed in this project. First, a review of the existing literature pertaining to InfoSec in agile development was conducted in order to inform discussion of alignments and disparities between the current body of knowledge and the state of practice as discovered through an empirical study (Strauss and Corbin, 1990). The literature used for this review primarily consists of peer-reviewed journal articles and conference papers from the IS and computer science domains. This review followed the systematic literature review method outlined by Kitchenham (Kitchenham, 2004, 2007) and Siddaway (2014), with key concepts from the literature being categorised and recorded in a concept matrix (Webster and Watson, 2002). Key search terms were identified from the research questions and conceptual foundations, which were then used to conduct repeated searches through electronic databases for potentially relevant literature. The identified articles were then reviewed in more detail for relevance, with only articles published after 2001 being included, and that explicitly discussed both InfoSec and agile development. The 2001 cut-off was chosen as this was the year that the Agile Manifesto was initially published. Exceptions were made for articles published prior to 2001 that provided foundational knowledge on a concept, though more recent articles were favoured where possible. The literature must pertain to organisational InfoSec, with consumer security and ethics of privacy being considered outside of the research scope. Once relevant literature was assessed for inclusion, the core concepts relating to InfoSec in agile development were categorised in a concept matrix, allowing the identification of related concepts for discussion in the literature review.

The authors used industry contacts to assist in identifying suitable participants. Thirteen individuals from eleven New Zealand-based organisations agreed to participate in this study through qualitative interviews (Table 1). The organisations ranged in size and maturity, from small start-up firms to multinational enterprises, with development teams working with both internal and external customers. Our sample therefore comprises a range of typical organisations. While all participants were experienced in working in an ASD context, they held a diverse set of positions, including security managers, lead developers, project managers, and chief technology officers. The variety of organisations and positions allowed for the identification of a broader range of approaches employed by practitioners than would have been possible with a more focused study. It was also noted that most participant organisations adopted an agile-traditional hybrid approach rather than a pure agile approach, explaining that pure agile is difficult to implement in practice. As such, some of the findings will also hold a degree of relevance to traditional development methodologies.

ID	Organisation Type	Position	Experience
P01	eCommerce platform	Head of Platform Development and Architecture	10-19 years
P02	Education provider	Architecture and Security Manager	20+ years
P03	SaaS vendor	Chief Technology Officer	20+ years
P04	IT services	Security Manager	20+ years
P05	Financial	Development Lead	20+ years
P06	IT services	Project Manager	20+ years
P07	IT services	Digital Technologies Business Manager	20+ years
P08	Security software vendor	Technical Product Manager	5-9 years
P09	SaaS vendor	Chief Technical Officer	10-19 years
P10	Media	DevOps Engineer	10-19 years
P11	Media	Senior DevOps Engineer	20+ years
P12	SaaS vendor	Security Architect	5-9 years
P13	SaaS vendor	Chief Information Security Officer	20+ years

Table 1: Summary of Participants

The interviews were conducted in a semi-structured format, with a protocol of prepared questions and discussion topics used to instigate and guide responses from the participants. Questions sought to understand key security concerns held by the participants and how ASD is adopted at their organisation, and then explored how security is addressed in all facets of an ASD project. Points of interest were followed up by the first author to provide a greater depth of understanding. The semi-structured approach allowed participants to openly discuss the topic area without confining them to a pre-defined set of questions, which is essential to an exploratory research project. The interview protocol was amended for each subsequent interview according to the role held by the participant and areas of inquiry identified from earlier interviews. Upon the completion of each interview, the audio-recording was transcribed and sent to the participant to provide the opportunity to check for accuracy and make clarifications. Theoretical saturation was considered to be attained as the later interviews did not result in the identification of any new factors, and only served to provide additional examples of these in practice.

The data analysis followed the guidelines for qualitative data coding described by Strauss and Corbin (1990), which outlines a three-stage process resulting in categories that describe the phenomenon at a high-level: open coding, axial coding, and theoretical coding. The authors worked inductively through the interview transcriptions line-by-line, identifying and extracting any valuable information, and then assigning it a concise descriptor, or open code. Axial coding was then performed by forming connections between the identified open codes; connections may relate to context, actions, or consequences. Any open codes determined to be irrelevant to the core phenomenon being investigated were discarded at this point. The process of identifying related open codes and collating them together produced concepts; a mid-level representation of the research findings. In theoretical coding, these concepts were finally related back to the core phenomenon under investigation, and categories are identified that explain the phenomenon at a high-level, creating a resultant representation of categorised CSFs described in section 4.

4 InfoSec in ASD: Perspectives from the Literature

This section provides a review of current literature pertaining to the inclusion of InfoSec considerations in ASD. Five focal areas are identified, for each area an overview of individual practices and solutions to addressing InfoSec is provided. The approaches identified here are later discussed in section 6 in relation to the current state of practice as discovered through empirical research and documented in section 5.

4.1 Agile Methodologies

Extreme Programming (XP), historically one of the more widely adopted agile methodologies, has in-built provisions for enabling the security of an IS (VersionOne and APLN, 2007; Wells, 1999). These provisions centre primarily around testing procedures, with automated unit testing of code ensuring a fundamental level of security (Wells, 1997). XP states that when a bug or vulnerability is found, new automatic tests are written to identify the same bug if it were to reoccur. Until the code has been corrected and retested, it cannot be released to the product. Academics have complemented these testing procedures with further adaptations to better integrate security-related needs in the project. Ge, Paige, Polack, and Brooke (2007) propose that specialised security training should be enforced for all project team members and stakeholders. They also recommend that the security architecture for a system should outline the relationship between system mechanisms and potential risks or vulnerabilities before development begins. Complementary to this, Boström et al. (2006) provide seven planning stages that can be utilised to address security. These stages include identifying critical assets that are unique to the system, techniques for security requirements representation and negotiation, and defining coding standards that the development team should adhere to.

As Scrum consistently has the highest adoption rate among agile teams, it has been the centre of security-oriented enhancements in the literature. Azham, Ghani, and Ithnin (2011) propose a dedicated Security Master role, who is responsible for ensuring the prioritisation and implementation of a security backlog. The security backlog is a prioritised list of security requirements alongside the traditional product backlog. The inclusion of a security analysis work spike early in each iteration, where the product backlog is updated with security requirements will sustain the visibility of security issues that must be addressed (Mougouei et al., 2013; Pohl and Hof, 2015). In alignment with the agile principle of work simplification, Pohl and Hof (2015) discourage the over-engineering of security by understanding the acceptable risk level for each project, rather than trying to provide a complete protection to all systems that may not require it (Beck et al., 2001). Despite agile development being a proponent of cross-functional teams, it is essential that developers recognise when external specialisation is needed to address security and are empowered to access these resources.

Microsoft has compiled and published an internal Security Development Lifecycle to provide best practices for assuring security in software development projects. Recognising the growing popularity of agile methodologies, Howard and Lipner (2006) provide recommendations for supplementing agile methodologies with practices from Microsoft's Security Development Lifecycle. These suggestions include integrating security as a measure of user story completeness, keeping code simple and releasing in small iterations to reduce the likelihood of unidentified vulnerabilities, identifying critical security skills and focusing these towards areas of high importance, and continually collaborating both within the team and with the project customer.

4.2 Strategies and Relationships

New IS strategies should be oriented to support agile security assurance, by promoting self-organisation, adaptability and flexibility of security development, both at a project and organisational level, as well as promoting innovation and improvement of an organisation's security capabilities (Dove, 2010). Strategies should be able to evolve with minimal resistance in order to respond to the ever-changing threat landscape (Dove, 2011). Compliance with such strategies and policies can be driven promoting awareness of InfoSec concerns, providing developers with necessary skills, fostering a team culture about the importance of InfoSec compliance, and encouraging open discourse about security concerns (Bartsch, 2011; Bulgurcu et al., 2010; Keramati and Mirian-Hosseinabadi, 2008; Weir et al., 2017). Not only is it important to have developers engaged in the secure development of software, but stakeholders and customers should also be involved (Bartsch, 2011). By regularly engaging in discourse with the customer throughout the project lifecycle, the project team can refine their understanding of the stakeholder's security requirements, even when the customer does not explicitly know these themselves.

4.3 Security Training

While the importance of having all project team members trained in secure development is recognised (Azham et al., 2011; Bartsch, 2011; Singhal and Singhal, 2011), research has shown that this does not translate into practice, with few practitioners being offered formal training by employers (Aguda, 2016; Bartsch, 2011). For those practitioners who do receive training, this usually takes the form of informal training through experience working with senior developers or methods to increase awareness of security concerns (Bartsch, 2011; Keramati and Mirian-Hosseinabadi, 2008).

All members of an agile development team should receive training and education in identifying and eliminating security vulnerabilities (Howard and Lipner, 2006). Ge et al. (2007) extend this proposal by suggesting that project stakeholders should also receive security training that is tailored to their individual responsibilities. By ensuring a baseline understanding and awareness of security across all people involved in an IS project, it is possible to minimise the introduction of vulnerabilities at all stages of the development lifecycle.

4.4 Requirements Management

Requirements engineering processes in agile methodologies emphasise user-oriented processes under typical operating conditions (Beck et al., 2001; Ramesh et al., 2010). Security incidents are an instance of abnormal events and thus are poorly captured during requirements elicitation and documentation (Boehm and Turner, 2005; Ramesh et al., 2010). As accurate and relevant, yet easily understood and implemented requirements are recognised as critical to the success of a project, solutions to integrating security into agile requirements engineering have emerged (Hofmann and Lehner, 2001; Siponen et al., 2005; Tappenden et al., 2006).

Abuser stories provide an extension of the widely adopted user story technique, offering agile teams a familiar approach that represents non-functional security requirements. Abuser stories describe the actions taken by individuals during a security incident, typically written from the perspective of either a user or an attacker (Boström et al., 2006; Kongsli, 2006; Peeters, 2005; Singhal and Singhal, 2011; Tondel et al., 2008). Misuse cases describe the sequence of events that occur during a security incident, providing developers with a representation of the entire process, and allowing countermeasures to be developed that address vulnerabilities exploited throughout an incident (Baca and Carlsson, 2011; Sindre and Opdahl, 2005; Siponen et al., 2005; Tondel et al., 2008). Despite the above recommendations, Aguda (2016) found that few specialised security requirements engineering techniques are used in agile contexts.

4.5 Security Testing

While testing is a long-recognised method of assessing the security of an application, it traditionally occurs late in the development lifecycle (Boehm, 1988). Due to the iterative delivery of agile projects, testing only towards the project conclusion is insufficient, and so practitioners conduct tests of the code on a regular basis using automated testing at all architectural layers (Baca and Carlsson, 2011; Tappenden et al., 2006; Wells, 1997). Regular automated testing allows early identification of vulnerabilities in the information system, so that these can be addressed and retested in subsequent iterations. Recommendations also include the inclusion of penetration and acceptance testing methods in early iterations of the development lifecycle (Kongsli, 2006; Singhal and Singhal, 2011). By conducting extensive manual tests early, major flaws in the code, architecture, and design of the system can be identified and resolved.

During testing phases, misuse cases should be prioritised according to the importance of the requirement and tested in order of priority to ensure that critical security needs are resolved first (Siponen et al., 2007, 2005). Further, it is recommended that simple and automated testing be used, as traditional testing approaches such as red team, penetration, and fuzzing are costly and time-consuming processes to conduct at each development iteration (Baca and Carlsson, 2011; Lipner, 2010; Tappenden et al., 2006). Berg and Ambler (2006) raise concerns that as security testing assesses negative requirements, there are too many possibilities that must be considered for comprehensive coverage. Instead, testers should randomly test a selection of inputs that provide sufficient, but not complete, testing coverage.

Despite the growing popularity of automated testing processes, Kongsli (2006) advises that manual testing should still be employed under specific circumstances. Some misuse cases are too complex to be adequately automated, and automatic testing is limited in its comprehensiveness compared to manual testing. Nassiri et al. (2012) argue that automated testing can be enhanced to manage complexity by testing each component of a system individually and then testing again as an integrated whole.

5 Critical Success Factors for Agile Security

The qualitative data analysis revealed twelve concepts as critical success factors across four categories: organisational practices, team practices, project practices, and technical practices (Figure 1). Each category details a set of related practices or solutions employed by practitioners to effectively address InfoSec during agile development projects. The four categories group the twelve concepts according to the level of granularity of the practices and the organisational level they are employed at. Promoting the importance of InfoSec and employing specific practices to address security in each of these four organizational areas fosters the delivery of InfoSec in a product. Each of the twelve concepts is representative of a critical success factor; activities or processes fundamentally involved in the delivery of secure information systems. For each of the twelve concepts, there are a multiplicity of practices and solutions that facilitate InfoSec assurance in the context of ASD. The specifics of these practices will vary from organisation to organisation, depending upon the culture, business model, and innumerable other factors. In the remainder of this section, we describe each concept. Whenever we refer to specific interviews, we refer to them by their participant ID in Table 1 in brackets (e.g., P01).

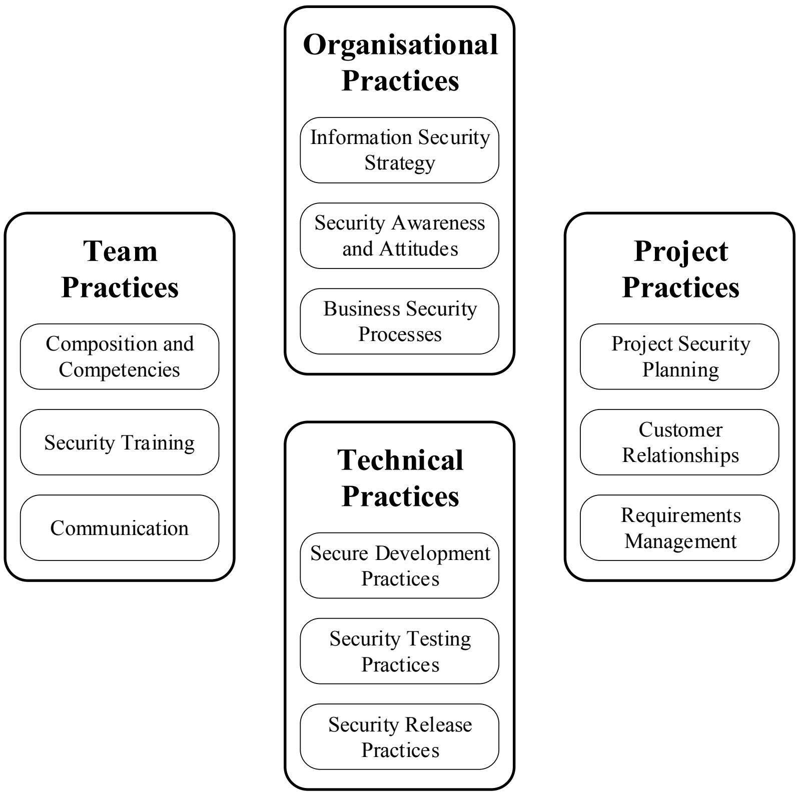
Organisational Practices
Information Security Strategy
Security Awareness and Attitudes
Business Security Processes
Team Practices
Composition and Competencies
Security Training
Communication
Technical Practices
Secure Development Practices
Security Testing Practices
Security Release Practices
Project Practices
Project Security Planning
Customer Relationships
Requirements Management

Figure 1: Categorised set of Agile Security CSFs

5.1 Organisational Practices

The organisational practices category encompasses solutions applied at an organisation-wide level. These practices impact all employees involved in the delivery of an IS and may extend to impact employees who are not directly involved in software delivery. Such people include organisational management, who are in the position to influence software development, even if indirectly, and other business units that may interact with software development teams. The following three concepts were identified as CSFs on the organisational level:

Information Security Strategy. An organisation should uphold strategic-level standards and expectations that apply across all teams and projects in order to enforce a general level of security that is deemed appropriate by senior leadership. Where possible, organisations should adapt industry standards to suit their internal characteristics, rather than trying to strictly adhere to a generalised process or rule set (P02, P09, P12). New best practices are also identified from conferences (P08) and more informally through the experience of staff and the organisation.

Security Awareness and Attitudes. The standards and expectations should be reflected in an adequate individual security awareness and attitude on the managerial and the developer level. Responsibility for security starts at the top, with senior leadership and executive board members having a vested interest in security, and an appreciation for the implications of security risks (P01, P09, P12). Individuals with the autonomy to make security-related decisions and recommendations, raise concerns, and take action where appropriate, will contribute significantly to the embedded security of the product (P03, P07, P09, P11, P10, P13). Formal processes such as peer reviews and explicit security requirements could even be supplanted by trust in staff that they will code defensively and know through experience where security issues may lie (P03, P04, P05, P09). The awareness and attitude should also extend to a collective level so that team members are being notified of any arising security concerns and acknowledging the importance of InfoSec to work being undertaken (P02, P03, P05, P06, P08, P11). This attitude towards security should be shared within and across teams to encourage team members to consider security in all of their tasks (P07, P10, P11, P13).

"…every single person is different, and then when you connect say, four team members together, and it's what, four to the power of four interactions, and there's a limit you just can't go past. So, you have to start with that, 'how do I make that person want to care about these things?'." - P11, Senior DevOps Engineer

Business Security Processes. As security is impacted by many units and teams within an organisation, processes that dictate and support these interactions are observed, especially in larger organisations. Among these are formal approval processes executed by a 'security council' (P01), internal consulting processes in a security-related community of practice (P07), or institutionalised re-use of past experiences through a repository of known issues and effective solutions (P12).

5.2 Team Practices

Team practices are those that relate to project teams responsible for software delivery and the individuals that comprise these teams. Practices in this category are specific to each team rather than being specific to the project being undertaken and may vary from team to team within a single organisation. Agile teams use their autonomy to determine approaches that are best suited to their dynamic. The following three concepts were identified as CSFs on the team level:

Composition and Competencies. Many participants identified a baseline level of security skills and awareness within agile teams to be essential to embedding security into a product. Some organisations (P01, P13) found that having a baseline level of security skills across all team members is sufficient to address their security needs, while other organisations with more extensive security needs complement teams with security specialists (P06). For small organisations, it may not be possible to source scarce security specialists. To address this issue, one participant identified the use of a 'bug bounty' program (P09), allowing the organisation to access freelance security skills to identify vulnerabilities that can then be resolved through internal baseline skills.

Security Training. The security skill level of agile team members will not always be sufficient to address security needs. In such situations, organisations often offer training programs or other opportunities to encourage the enhancement of developer skill sets. These opportunities range from relatively informal offerings such as mentoring (P09) or experience sharing through storytelling to more formal approaches such as 'hack-days' (P13, see quote below) or group-led seminars relating to specific security techniques and online videos to provide annual refreshers on industry standards (P01, P08).

"We go through examples of insecure development practices, and how they can avoid that. The best thing is that those examples are using our own application, so we create versions for the dev [sic] training that contain development mistakes that affect security…" - P13, Chief Information Security Officer

Recognising the importance of the customer in agile projects, one participant (P04) noted that providing security training to product owners was essential to being able to understand the security needs of the customer accurately.

Communication. Participants frequently reported that InfoSec is not a regular discussion topic at the daily stand-ups, instead only being raised when a current user story has an explicit security requirement (P04, P07, P10, P13). Instead, less formal alternatives to communication are adopted by agile development teams, both face-to-face and virtual through mailing lists (P08) or digital tools such as Slack (P01, P08, P12, P13), sometimes with dedicated security channels (P12).

"We have a few Slack channels ranging from 'do you need help with security?'… and the other is just a pure interest channel, anything security related, just to raise awareness and keep it light-hearted and fun." - P12, Security Architect

5.3 Project Practices

This category includes practices that are employed at the level of individual projects and may vary depending upon the characteristics of each project. The customer and the team will both be influential in determining which practices should be employed in this category. Approaches taken here will set standards for how the security needs of the customer will be integrated into the lifecycle of an ASD project. The following three concepts were identified as CSFs on the project level:

Project Security Planning. Participants commonly acknowledged that security cannot easily be addressed in an ad hoc manner; project teams must identify notable security risks as early as possible so that they can be given appropriate consideration and solutions can be identified. This process includes both initial risk identification at the outset of a project (P01, P02, P05, P08), and for each sprint or work iteration (P03). The teams should also have processes for determining the priority of newly arisen security requirements or concerns and be able to adjust the planned workflow to address them (P09, P13).

> *"In the planning phase we start to think, 'how can people either abuse this on purpose, or mess it up by accident?' And we need to address those things." - P03, Chief Technology Officer*

Customer Involvement. Involving the customer was reported to be critical to achieving a satisfactory level of security, as they are ultimately responsible for determining what level of security risk is considered acceptable, and how much to spend on risk mitigation and countermeasures (P07, P08). Involving the customer both early and continually in discussions pertaining to security and clarifying security expectations in planning documentation helps ensure that the system successfully meets the customer's security needs (P02; P07). In some instances, they lack the expertise to understand their security needs fully. In such cases, the project team must work closely with the customer to help develop their understanding of the associated risks and identify what countermeasures will be necessary (P04, P06, P07).

> *"A typical customer will say they want the software to be secure. 'Well, how secure do you want it to be?'... 'We don't know'. So, they'll just write the contract to say be secure. We get that requirement upfront in the agile project and have to decompose it a bit." - P04, Security Mgr.*

Security Requirements Management. Several techniques for ongoing management of security-related requirements have been adopted by practitioners, with many of these being adaptations of existing agile practices. Misuse cases and abuser stories were both identified as successful practices (P04, P12), where security requirements are written from the perspective of a user or an attacker. More common was the integration of security into regular user stories through a definition of done or completeness checklist (P03, P08, P09), or by relating the story to another document such as a security matrix or information flow diagram to identify how new functionality will relate to the security of the product (P07, P13).

5.4 Technical Practices

The most granular practices are those relating to the technical implementation of security in the focal product. These concepts describe techniques used to embed security into the deliverable itself and provide assurance that this has been performed to an acceptable standard. The following three concepts were identified as CSFs as technical practices:

Secure Development Practices. Participants identified development practices that can be used to support the implementation of security into the product codebase, with many of these being derived from existing agile practices. Developers who collaborate tend to produce more secure code, which can be achieved through pair and mob-programming or peer code reviews (P01, P02, P04, P08, P11). These practices encourage developers to work in groups, providing a second perspective on security issues and acting as an instantaneous peer review of the code.

> *"We try to do a mixture of peer-programming together; mobbing, where it's more than two people for example. If it's a singleton, so one person working on the code, instead of them just pushing it up the way of the people who review it, they'll spend an hour with us just walking us through, so we can ask questions on the spot." - P11, Senior DevOps Engineer*

Agile security can be further supported through the appropriate structuring of work. Keeping the amount of work completed in each pull-request to a minimum facilitates faster peer-reviews and reduces the likelihood of a vulnerability being undetected (P11). When a new security concern arises during the development of a deliverable, developers should attach the new security task as a sub-task of the primary deliverable. As agile promotes delivery of functional requirements, tying unexpected non-functional requirements to a deliverable will 'pull' the security requirement through development, ensuring security requirements are addressed (P12).

Security Testing Practices. Before release, the security of deliverables should be validated. While this assurance is partially provided through peer reviews, formal testing is more comprehensive and rigorous. As testing is typically very process-oriented and time-consuming, participants identified the importance of 'shifting security left', i.e. conducting tests earlier to provide time to address any issues identified. Additionally, keeping testing processes simple and small reduces the impediment of testing to agility (P08, P12), achieved through automated testing tools which provide a baseline level of testing coverage (P03, P05, P08, P11). These automated testing tools may need to be supplemented by manual procedures for more complex security requirements (P12). Choosing critical areas to focus testing on rather than full system coverage will also aid in streamlining the security testing process.

> *"We could do one hundred percent coverage, but that's usually a waste of time because you're almost certainly not testing the things that you really need to test." - P11, Senior DevOps Engineer*

Penetration testing is a common practice for identifying vulnerabilities in an IS, though this is a costly and time-consuming process. Due to these constraints, participants suggest that penetration testing is only used at major milestones, such as before a critical deliverable (P04, P08). Acceptance tests typically assess whether a function performs its intended tasks (P04); inverting this test can validate if the function will continue to perform when abused, such as bad data input, thereby identifying potential vulnerabilities without the need for developing a comprehensive testing security protocol from the ground up.

Security Release Practices. Once the security of a deliverable has been verified in a staging environment, it can be released into production. The shift in environment and potential for vulnerabilities to be introduced into production necessitates further security assurances practices. Simplifying the process through automation and pattern identification for successful release of minor deliverables speeds up the release process while reducing the risk of operator-introduced errors, where an unexpected change in the architecture of production systems may introduce a new vulnerability (P01, P03, P08, P11). Different approaches to release deadline conflicts arising from security issues were observed. Some organisations operate a continuous integration/continuous deployment model, with no scheduled releases, in which security concerns can be comprehensively addressed without delaying a release (P10). For teams with a fixed release schedule, a choice must be made to either push-back the delivery date, or to release insecure code, with the former usually taking precedence (P10).

6 Discussion

Answering RQ2, our main theoretical contribution is a set of twelve categorised CSFs (Figure 1) for the assurance of InfoSec in an ASD context that is more comprehensive and grounded in actual practice than existing literature. These categorised CSFs provide a representation of agile security practices that encompasses a broad range of focus areas and highlight the importance of practices at a high level that enable individuals, engage management, and develop a positive attitude towards security. When we contrast our findings with the five focal themes we identified in the literature while addressing RQ1 (agile methodologies, strategies and relationships, security training, requirements management, and security testing) to answer RQ3, we find that the existing literature focusses primarily on techniques and processes that can be used by teams to embed security into the development of an IS. Recommendations regarding strategy, culture, and people are generally high-level statements indicating the importance of these areas but provide few actionable recommendations. In contrast to this emphasis on techniques and processes, practitioners were observed to place heavy emphasis on the importance of people, their interactions, and project coordination, rather than specific techniques, in alignment with the agile manifesto (Beck et al., 2001). We also expand upon previous suggestions of the importance of security training and awareness, by identifying specific approaches taken by practitioners to address these concerns, including informal mentorship and practical training sessions.

Note that we were not able to identify any approaches employed in practice that fit into the first of the literature themes: agile methodologies. Practitioners do not appear to select agile methodologies on the basis of their ability to address security concerns. Their choice of a methodology is primarily guided by the methodology's fit in the team and organisational dynamic and its ability to facilitate rapid delivery of a quality outcome. The reasoning for these practices not being employed in practice is unclear, though this is perhaps attributable to either a lack of visibility of these proposed methodologies or being considered unsupportive of primary development goals.

For the other four themes identified in the literature review that we could observe in the findings of this study, we found subtle variations in the specific solutions employed in practice when compared to academic recommendations. Recommendations in the literature including early and automated testing for security vulnerabilities and the use of abuser stories and misuse cases to represent security requirements were observed to be employed in practice. However, practices from the literature such as security backlogs and early penetration testing were not identified by participants, indicating a gap between existing recommendations and the actual practice in ASD.

Our identified CSFs also have practical implications. They provide a comprehensive foundation for teams and organisations to assess and further develop their internal best practices. We posit that by addressing these twelve CSFs through the use of appropriate practices, ASD teams can contribute to mitigating the risks involved in InfoSec assurance, while retaining benefits that arise from agile development, such as rapid delivery, adaptability to change, and improved alignment with the customer's requirements.

However, our research is not without limitations. The geographic as well as the role diversity of participants is restricted; all participants work in New Zealand, and many roles involved in agile development projects, such as product owners and Scrum masters, are not represented. A broader scope of participants may reveal that the practices identified in the literature that we did not find to be employed in practice were merely not observed due to the size of the data collected for this study. Additionally, the qualitative nature of this research and the use of semi-structured interviews for data collection and inductive coding for data analysis presents a further limitation. The information provided by participants is subject to cognitive bias introduced by their personal experiences, interests, and opinions. The role of the researcher in analysing the collected data may also introduce bias through the interpretation of the data, despite following established guidelines for coding to reduce the risk of the researcher's interpretation impacting the analysis.

7 Conclusion

We developed a set of twelve CSFs, which identifies activities and processes where organisations and teams must focus their efforts through strategically selected best practices to assure the embedment of InfoSec in ASD projects. These CSFs are categorised into four categories: organisational, team, project, and technical. While identified as imperative to reducing the InfoSec risk associated with software produced according to ASD principles, the application of these CSFs is flexible, allowing development teams to strategically identify and adopt security practices that are aligned with team and organisational dynamics and priorities. While these CSFs were developed with the intent of addressing tension between InfoSec and ASD, they may also be of value to traditional development models, though requiring different specific practices in order to be most effectively implemented.

While these CSFs were identified inductively from data that reflects practice, they currently nevertheless remain conceptual. Future work should therefore attempt to validate these constructs, find relationships between them, and evidence for their effectiveness at assuring secure ASD. Future research can also expand the data collection further across organisations in different countries, of different sizes, maturity levels, industries, or business models. Subsequently, the presented CSFs can inform the development of recommendations that comprehensively address agile security concerns, by considering which critical success factors have underdeveloped recommendations and require further attention. Further research can also design and evaluate suitable metrics for the CSFs. This study also highlights the complexity of the tension between InfoSec and ASD, and identifies the need for further exploration into the nature of this tension and why it may not easily be resolved.

References

Acar, Y., Stransky, C., Wermke, D., Weir, C., Mazurek, M. L., and Fahl, S. (2017). "Developers Need Support, Too: A Survey of Security Advice for Software Developers." In *2017 IEEE Cybersecurity Development (SecDev)* (pp. 22–26).

Aguda, O. A. (2016). Effectiveness of security requirements engineering in agile/scrum software development projects (D.C.S.). Colorado Technical University, United States -- Colorado.

Andress, J. (2014). "What is Information Security?" In *The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice* (2nd ed., pp. 1–22). Rockland, MA, UNITED STATES: William Andrew.

Australian Computer Society. (2016). "Cybersecurity: Threats, Challenges, Opportunities."

Azham, Z., Ghani, I., and Ithnin, N. (2011). "Security backlog in Scrum security practices." In *2011 Malaysian Conference in Software Engineering* (pp. 414–417).

Baca, D., and Carlsson, B. (2011). "Agile development with security engineering activities." In *Proceeding of the 2nd workshop on Software engineering for sensor network applications - SESENA '11* (pp. 149–258). Waikiki, Honolulu, HI, USA: ACM Press.

Bagiński, J., and Rostański, M. (2011). "The modeling of business impact analysis for the loss of integrity, confidentiality and availability in business processes and data." *Theoretical and Applied Informatics* 23 (1), 73–82.

Bartsch, S. (2011). "Practitioners' Perspectives on Security in Agile Development." In *2011 Sixth International Conference on Availability, Reliability and Security* (pp. 479–484).

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). "Manifesto for Agile Software Development."

Bellovin, S. (2015). *Thinking Security: Stopping Next Year's Hackers* (1st edition.). Addison-Wesley Professional.

Berg, C., and Ambler, S. W. (2006). "Assurance & Agile Processes." *Dr. Dobb's Journal; San Mateo* 31 (7), 42–45.

Bernard, T. S., and Cowley, S. (2017). "Equifax Breach Caused by Lone Employee's Error, Former C.E.O. Says - The New York Times."

Boehm, B. (1988). "A spiral model of software development and enhancement." *Computer* 21 (5), 61–72.

Boehm, B., and Turner, R. (2005). "Management Challenges to Implementing Agile Processes in Traditional Development Organizations." *IEEE Software; Los Alamitos* 22 (5), 30–39.

Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K., and Kruchten, P. (2006). "Extending XP Practices to Support Security Requirements Engineering." In *Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems* (pp. 11–18). New York, NY, USA: ACM.

Boynton, A. C., and Zmud, R. W. (1987). "An Assessment of Critical Success Factors." *Sloan Management Review* 25 (4), 17–27.

Bulgurcu, B., Cavusoglu, H., and Benbasat, I. (2010). "Information Security Policy Compliance: An Empirical Study of Rationality-Based Beliefs and Information Security Awareness." *MIS Quarterly* 34 (3), 523–548.

Chau, T., and Maurer, F. (2004). "Knowledge Sharing in Agile Software Teams." In W. Lenski (Ed.), *Logic versus Approximation: Essays Dedicated to Michael M. Richter on the Occasion of his 65th Birthday* (pp. 173–183). Berlin, Heidelberg: Springer Berlin Heidelberg.

Chen, D. Q., Mocker, M., Preston, D. S., and Teubner, A. (2010). "Information Systems Strategy: Reconceptualization, Measurement, and Implications." *MIS Quarterly* 34 (2), 233–259.

Choo, K.-K. R. (2011). "The cyber threat landscape: Challenges and future research directions." *Computers & Security* 30 (8), 719–731.

Chung, L., and do Prado Leite, J. C. S. (2009). "On Non-Functional Requirements in Software Engineering." In A. T. Borgida, V. K. Chaudhri, P. Giorgini, and E. S. Yu (Eds.), *Conceptual Modeling: Foundations and Applications: Essays in Honor of John Mylopoulos* (pp. 363–379). Berlin, Heidelberg: Springer Berlin Heidelberg.

Collins, L. (2013). “System Security.” In J. R. Vacca (Ed.), *Cyber Security and IT Infrastructure Protection* (pp. 233–246). Rockland, MA, UNITED STATES: William Andrew.

Curtis, B., Sappidi, J., and Szynkarski, A. (2012). “Estimating the size, cost, and types of Technical Debt.” In *2012 Third International Workshop on Managing Technical Debt (MTD)* (pp. 49–53). Zurich, Switzerland: IEEE.

Dingsøyr, T., and Dybå, T. (2010). *Agile Software Development Current Research and Future Directions*. Berlin, Heidelberg: Springer Berlin Heidelberg.

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B. (2012). “A decade of agile methodologies: Towards explaining agile software development.” *Journal of Systems and Software* 85 (6), 1213–1221.

Dove, R. (2010). “Pattern qualifications and examples of next-generation agile system-security strategies.” In *44th Annual 2010 IEEE International Carnahan Conference on Security Technology* (pp. 902–908).

Dove, R. (2011). “Patterns of Self-Organizing Agile Security for Resilient Network Situational Awareness and Sensemaking.” In *2011 Eighth International Conference on Information Technology: New Generations* (pp. 902–908).

Dynes, S., Goetz, E., and Freeman, M. (2008). “Cyber Security: Are Economic Incentives Adequate?” In E. Goetz and S. Shenoi (Eds.), *Critical Infrastructure Protection* (pp. 15–27). Springer US.

Elbanna, A., and Sarker, S. (2016). “The Risks of Agile Software Development: Learning from Adopters.” *IEEE Software* 33 (5), 72–79.

Ellis, S. R. (2013). “Fundamentals of Cryptography.” In J. R. Vacca (Ed.), *Cyber Security and IT Infrastructure Protection* (pp. 295–308). Rockland, MA, UNITED STATES: William Andrew.

Freund, Y., P. (1988). “Critical Success Factors.” *Planning Review* 16 (4), 20–23.

Ge, X., Paige, R. F., Polack, F., and Brooke, P. (2007). “Extreme Programming Security Practices.” In G. Concas, E. Damiani, M. Scotto, and G. Succi (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 226–230). Springer Berlin Heidelberg.

Glass, R. L. (2001). “Agile Versus Traditional: Make Love, Not War.” *Cutter IT Journal* 14 (12), 72–79.

Glinz, M. (2007). “On Non-Functional Requirements.” In *15th IEEE International Requirements Engineering Conference (RE 2007)* (pp. 21–26).

Hofmann, H. F., and Lehner, F. (2001). “Requirements engineering as a success factor in software projects.” *IEEE Software; Los Alamitos* 18 (4), 58–66.

Höne, K., and Eloff, J. H. P. (2002). “Information security policy — what do international information security standards say?” *Computers & Security* 21 (5), 402–409.

Hood, A. (2017). *Security Certification or How I Learned to Stop Worrying & Love Stories*.

Howard, M., and Lipner, S. (2006). “Integrating SDL with Agile Methods.” In *The Security Development Lifecycle* (pp. 225–240). Redmond, WA, USA: Microsoft Press.

Kang, A. N., Barolli, L., Park, J. H., and Jeong, Y.-S. (2014). “A strengthening plan for enterprise information security based on cloud computing.” *Cluster Computing* 17 (3), 703–710.

Keramati, H., and Mirian-Hosseinabadi, S. (2008). “Integrating software development security activities with agile methodologies.” In *2008 IEEE/ACS International Conference on Computer Systems and Applications* (pp. 749–754).

Kitchenham, B. (2004). “Procedures for performing systematic reviews.” *Keele, UK, Keele University* 33, 2004.

Kitchenham, B. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, EBSE Technical Report EBSE-2007-01.

Kongsli, V. (2006). “Towards Agile Security in Web Applications.” In *Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications* (pp. 805–808). New York, NY, USA: ACM.

Kropp, M., Meier, A., Anslow, C., and Biddle, R. (2018). “Satisfaction, practices, and influences in agile software development.” In *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018* (pp. 112–121). ACM.

Leidecker, J. K., and Bruno, A. V. (1984). "Identifying and using critical success factors." *Long Range Planning* 17 (1), 23–32.

Licorish, S. A., Holvitie, J., Hyrynsalmi, S., Leppänen, V., Spínola, R. O., Mendes, T. S., … Buchan, J. (2016). "Adoption and Suitability of Software Development Methods and Practices." In *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)* (pp. 369–372).

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., … Zelkowitz, M. (2002). "Empirical Findings in Agile Methods." In D. Wells and L. Williams (Eds.), *Extreme Programming and Agile Methods — XP/Agile Universe 2002* (pp. 197–207). Springer Berlin Heidelberg.

Lipner, S. (2010). "Security development lifecycle." *Datenschutz und Datensicherheit - DuD* 34 (3), 135–137.

McGraw, G. (2006). *Software Security: Building Security In* (1st edition.). Addison-Wesley Professional.

Mougouei, D., Sani, N. F. M., and Almasi, M. M. (2013). "S-Scrum a Secure Methodology for Agile Development of Web Services." *World of Computer Science and Information Technology Journal* 3 (1), 15–19.

Nassiri, R., Janeh, H., and Jabbehdari, S. (2012). "A Security Test Method on Agile Software Development." *International Journal of Advanced Research in Computer Science; Udaipur* 3 (1), 154–157.

Peeters, J. (2005). "Agile Security Requirements Engineering." In *SREIS 2005*. Paris, France.

Peppard, J., and Ward, J. (2004). "Beyond strategic information systems: towards an IS capability." *The Journal of Strategic Information Systems* 13 (2), 167–194.

Petersen, K., and Wohlin, C. (2009). "A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case." *Journal of Systems and Software* 82 (9), 1479–1490.

Pohl, C., and Hof, H.-J. (2015). "Secure Scrum: Development of Secure Software with Scrum."

Ramesh, B., Cao, L., and Baskerville, R. (2010). "Agile requirements engineering practices and challenges: an empirical study." *Information Systems Journal* 20 (5), 449–480.

Rector, K. (2018). "Hack of Baltimore's 911 dispatch system was ransomware attack, city officials say." *baltimoresun.com*.

Siddaway, A. (2014). "What is a Systematic Literature Review and How Do I Do One?"

Sindre, G., and Opdahl, A. L. (2005). "Eliciting security requirements with misuse cases." *Requirements Engineering* 10 (1), 34–44.

Singhal, S., and Singhal, A. (2011). "Development of Agile Security Framework Using a Hybrid Technique for Requirements Elicitation." In S. Unnikrishnan, S. Surve, and D. Bhoir (Eds.), *Advances in Computing, Communication and Control* (pp. 178–188). Springer Berlin Heidelberg.

Siponen, M. (2006). "Information Security Standards Focus on the Existence of Process, Not Its Content." *Commun. ACM* 49 (8), 97–100.

Siponen, M., Baskerville, R., and Kuivalainen, R. (2007). "Extending Security in Agile Software Development Methods." In H. Mouratidis and P. Giorgini (Eds.), *Integrating Security and Software Engineering: Advances and Future Visions* (pp. 144–159). IGI Global.

Siponen, M., Baskerville, R., and Kuivalainen, T. (2005). "Integrating security into agile development methods." In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences* (pp. 185–191).

Siponen, M., and Willison, R. (2009). "Information security management standards: Problems and solutions." *Information & Management* 46 (5), 267–270.

Strauss, A. L., and Corbin, J. M. (1990). *Basics of qualitative research: grounded theory procedures and techniques*. Newbury Park, Calif.: Sage Publications.

Tappenden, A. F., Huynh, T., Miller, J., Geras, A., and Smith, M. (2006). "Agile Development of Secure Web-Based Applications." *International Journal of Information Technology and Web Engineering (IJITWE)* 1 (2), 1–24.

Tondel, I. A., Jaatun, M. G., and Meland, P. H. (2008). "Security Requirements for the Rest of Us: A Survey." *IEEE Software* 25 (1), 20–27.

VersionOne, and APLN. (2007). "2nd Annual Survey 'The State of Agile Development.'"

VersionOne, and CollabNet. (2017). "The 12th Annual State of Agile Report."

von Solms, R., and van Niekerk, J. (2013). "From information security to cyber security." *Computers & Security* 38, 97–102.

Webster, J., and Watson, R. T. (2002). "Analyzing the Past to Prepare for the Future: Writing a Literature Review." *MIS Quarterly* 26 (2), xiii–xxiii.

Weir, C., Rashid, A., and Noble, J. (2017). "I'd Like to Have an Argument, Please: Using Dialectic for Effective App Security." In *Proceedings 2nd European Workshop on Usable Security*. Paris, France: Internet Society.

Wells, D. (1997). "Extreme Rules - When a Bug is Found."

Wells, D. (1999). "Extreme Programming Rules."